UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

DETECCIÓN DE PERSONAS UTILIZANDO CONTEXTO DE LA ESCENA

Carlos Chaparro Pozo
Tutor: Álvaro García Martín
Ponente: José M. Martínez Sánchez

Junio 2016

DETECCIÓN DE PERSONAS UTILIZANDO CONTEXTO DE LA ESCENA

Carlos Chaparro Pozo
Tutor: Álvaro García Martín
Ponente: José M. Martínez Sánchez

Video Processing and Understanding Lab
Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio 2016

Trabajo parcialmente financiado por el Ministerio de Economía y Competitividad del Gobierno de España bajo el proyecto TEC2014-53176-R (HAVideo) (2015-2017)

Resumen

En la actualidad, el procesamiento de imagen y vídeo digital está inmerso en una evolución constante. La utilidad del mismo es muy amplia y abarca muchos campos como, por ejemplo, la vídeo-seguridad. Esta es un área muy relevante y actual, entendiéndose la investigación vital para la mejora de las técnicas de análisis.

Dentro del campo de la vídeo-seguridad, la detección de personas es una de las tareas más complejas. La detección de personas afronta múltiples desafíos tales como el cambio de posición, la variación de iluminación o las oclusiones. Cuando alguna de las anteriores características está presente y el entorno es complejo, los algoritmos existentes reducen considerablemente su rendimiento. Por este motivo, este trabajo tiene como objetivo la implementación de un sistema que mejore el rendimiento y la funcionalidad de la detección de personas utilizando contexto de la escena.

En este momento, los modelos propuestos para solucionar los entornos con gran variabilidad y complejidad necesitan reentrenarse para ofrecer un buen rendimiento a un caso concreto. En este trabajo, hemos implementado un sistema en C++ que permite anotar las zonas oclusivas de la escena al usuario, evitando volver a entrenar al modelo y mejorando asimismo su funcionalidad y rendimiento. El algoritmo de detección de personas implementado es una adaptación del algoritmo conocido como DTDP (*Discriminatively Trained Part Based Models*). Este nos permite utilizar el contexto para mejorar la detección de personas en función de las partes oclusivas de la escena y de la relación entre las escalas de análisis y las ventanas de detección.

Para completar el objetivo se ha realizado una evaluación del sistema implementado, comparándolo con el detector base DTDP. De esta manera podemos comprobar que el sistema es más eficiente, además de la mejora respecto a la funcionalidad.

Palabras clave

Detección de personas, escala, mapas de confianza, marcadores, modelo, objetos estáticos, oclusión, rendimiento, segmentación y vídeo-seguridad.

Abstract

At present, video signal processing is immersed in a constant evolution. The scope of it is wide and includes many fields as, for example, video security. This one is a very relevant and current area. Therefore, research in this area is vital to improve the existing techniques of analysis.

In video security, people detection is one of the most complex tasks. People detection faces many challenges related to pose changes, illumination variations and occlusions. When some of the previous characteristics are present and the scene is complex, the existing algorithms reduce considerably its performance. For it, the aim of this project is the implementation of a framework that improves the performance and the functionality of people detection using context of the scene.

At this moment, the models proposed to cope the environments with great variability and complexity need to re-train to offer an optimal performance to a concrete case. In this project, we have implemented a framework in C ++ that allows to the users annotate the occlusive zones of the scene, avoiding to re-train the model and improving its functionality and performance. The algorithm implemented is an adaptation of the algorithm known as DTDP (*Discriminatively Trained Part Based Models*). This allows us to use the context to enhance the people detection related to the occlusive parts of the scene and the relation between the scales of analysis and the size of the detection windows.

In order to complete our aim, we realized an evaluation of the implemented framework, comparing it with DTDP detector. In this manner, we can verify that the framework is more efficient, besides the improvement regarding the functionality.

Keywords

Confidence maps, markers, model, occlusion, people detection, performance, scale, segmentation, static objects and video security.

Agradecimientos

En primer lugar quiero agradecer a mi tutor, Álvaro, por confiar en mí para llevar a cabo este trabajo. Su apoyo, motivación e implicación a lo largo de todo el proceso han sido vitales para que el esfuerzo realizado se materializase en todas estas páginas.

También quiero expresar mi gratitud a José María Martínez y a Jesús Bescós por haberme dado la oportunidad de realizar este trabajo.

Por último, agradecer al grupo de trabajo VPU su amabilidad, disposición para echar siempre una mano y el agradable ambiente de trabajo.

Carlos Chaparro Pozo

Junio 2016

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Motivación

En los últimos años, se han llevado a cabo enormes esfuerzos y progresos en el procesamiento de imagen y vídeo digital debido a su gran utilidad dentro de la sociedad en la que vivimos. Actualmente existe una gran demanda en el área de la seguridad, siendo la vídeo-vigilancia uno de los campos en los que más recursos hay destinados a la investigación.

Dentro del área de investigación en la vídeo-seguridad, existen una gran cantidad de algoritmos utilizados para diversas aplicaciones, ya sea detección de sucesos, objetos, personas, etc. La detección de personas pertenece a esta línea de investigación y actualmente hay múltiples detectores, en busca del rendimiento óptimo en casos complejos. La problemática de la detección de personas se encuentra, principalmente, en la dificultad para definir un modelo de las mismas, debido a la gran variabilidad en la apariencia física, poses, puntos de vista, movimiento e interacción entre diferentes personas y objetos.

A la hora de detectar personas, hay que tener en cuenta que en muchos casos las escenas presentan objetos estáticos que ocultan parte de las personas a detectar. Una secuencia puede tener zonas, como una mesa o un techo, que ocluyan parte de las personas del vídeo, denominándose zonas de oclusión. Esto implica una considerable reducción del rendimiento de la mayoría de los algoritmos, ya que habría que reentrenar el modelo de persona. Al tener en cuenta información de contexto, se pueden obtener resultados óptimos a la vez que mejorar la funcionalidad del detector.

Como resultado de estas reflexiones, la motivación principal de este trabajo es lograr la implementación de un sistema que detecte personas utilizando información de contexto, así como la creación de un algoritmo para anotar las zonas oclusivas

de la escena. La finalidad del trabajo sería obtener una solución práctica y efectiva al problema de detectar personas con partes ocluidas. Por último, se evaluará el rendimiento del sistema implementado comparándolo con el del algoritmo original, DTDP.

1.2. Objetivos

Una vez definida la motivación de este trabajo, establecemos los objetivos partiendo de propósitos parciales que finalmente permitirán alcanzar el objetivo global de una manera progresiva y adecuada. El desglose de los objetivos es el siguiente:

1. **Estudio del estado del arte**
   Análisis en profundidad de los algoritmos a utilizar para implementar las siguientes funcionalidades:

   - **Segmentación:** estudiar el algoritmo *Watershed* [1], el cual utilizaremos para anotar de manera sencilla e intuitiva las zonas oclusivas.
   - **Obtención de mapas de confianza de oclusión:** algoritmo propuesto en [2] consistente en adaptar la imagen con las zonas oclusivas anotadas, obtenida tras la segmentación, a los niveles de escala requeridos por el detector DTDP (*Discriminatively Trained Part Based Models*) [3].
   - **Detección de personas:** analizar un método de detección de personas basado en modelos de diferentes partes del cuerpo como el DTDP multiconfiguraciones [4], una variación del detector DTDP (*Discriminatively Trained Part Based Models*) [3], donde cada parte del cuerpo es modelada según el algoritmo HOG (*Histogram of Object Gradients*) [5]. Este detector se adaptará para utilizar los mapas de confianza de oclusión obtenidos en base al planteamiento de [2], permitiendo mejorar el rendimiento al utilizar información de contexto.

2. **Aprendizaje de herramientas y bibliotecas**
   Utilizar las herramientas y bibliotecas necesarias para el correcto desarrollo de este trabajo: OpenCV [6] y Matlab [7].

3. **Creación de un algoritmo para anotar zonas oclusivas**
   Implementar, tomando como base el algoritmo *Watershed* [1], una técnica que permita a los usuarios anotar de manera sencilla e intuitiva las zonas oclusivas de la escena para utilizar la información de contexto en la detección de personas.

4. **Adaptación y mejora del algoritmo de detección de personas DTDP**
   Implementación de un detector que utilice información de contexto en C++, tomando como base el código de los algoritmos estudiados en el estado del arte. Asimismo, añadir el algoritmo para anotación de las zonas oclusivas, mejorando la funcionalidad del sistema.

5. **Evaluación y análisis de los resultados**
   Comparar el rendimiento del sistema implementado respecto al algoritmo existente, permitiéndonos conocer sus ventajas y limitaciones.

Con todo ello la finalidad que se persigue no es otra que la de conseguir un sistema práctico y efectivo en C++ para el algoritmo presentado por [2], denominado DTDP-Context (*Discriminatively Trained Part Based Models - Context*).

1.3. Estructura de la memoria.

La memoria del proyecto se divide en los siguientes capítulos:

- **Capítulo 1. Introducción:** motivación y objetivos del proyecto.
- **Capítulo 2. Estado del arte:** estudio de las características y peculiaridades de cinco algoritmos concretos: *Watershed* [1], HOG [5], DTDP [3], DTDP multiconfiguraciones [4] y DTDP-Context [2].
- **Capítulo 3. Diseño y desarrollo:** descripción de la adaptación y programación del sistema de detección de personas utilizando contexto de la escena, compuesto por tres etapas diferenciadas.
- **Capítulo 4. Evaluación del sistema:** estudio del criterio de evaluación y análisis de los resultados obtenidos para el detector base DTDP y nuestro sistema de detección utilizando información de contexto.
- **Capítulo 5. Conclusiones y trabajo futuro:** conclusiones obtenidas tras el análisis de resultados del trabajo presentado. Trabajo pendiente y trabajo futuro.

Capítulo 2

Estado del arte

La detección de personas consiste principalmente en diseñar y entrenar un modelo de persona basado en parámetros característicos de las misma (dimensiones, silueta, etc); para después, ajustar este modelo a los posibles objetos de una escena y determinar si son personas o no. Solamente los objetos candidatos que se ajusten al modelo serán clasificados como una persona.

Uno de los factores más relevantes a la hora de detectar los candidatos que se ajusten al modelo es el contexto de la escena. La detección de personas en escenarios complejos es un problema que ha de tenerse en cuenta a la hora de plantear cualquier algoritmo de detección de personas. Existen algoritmos de detección como el DTDP [3] que comienzan a fallar con oclusiones de un 20 % entre objetos y que por encima de este porcentaje las detecciones se vuelven cada vez más inusuales.

Por este motivo, en este trabajo implementaremos un sistema en el que el usuario pueda anotar las zonas oclusivas de la escena, permitiendo al detector mejorar su rendimiento. Antes de estudiar los algoritmos utilizados, exponemos el planteamiento del sistema, lo que nos permitirá tener una visión general del mismo y facilitar la compresión del papel de cada técnica.

2.1. Planteamiento del sistema

A la hora de afrontar la problemática del trabajo, dividimos el diseño en tres etapas que componen en conjunto el sistema a implementar. Cada una de ellas tiene una función por separado, aunque en todo momento hay que tener en cuenta que el resultado de cada etapa sea coherente y adaptable a la siguiente. En la figura 2.1 podemos ver el esquema de las etapas, de las que resumiremos la funcionalidad y el motivo de la elección de los algoritmos para cada una de ellas. Las características

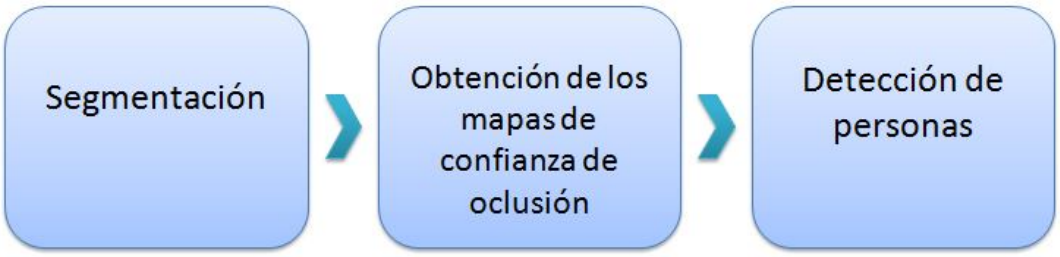


Figura 2.1: Etapas del sistema implementado. Primero segmentaremos la imagen y anotaremos las zonas oclusivas, obtendremos los mapas de confianza de oclusión y por último, los utilizaremos para mejorar el rendimiento de la detección de personas.

principales de cada etapa son:

- **Segmentación:** en la sección 2.2 veremos el algoritmo *Watershed* [1]. Esta es una técnica de segmentación morfológica que nos permite dividir en regiones imágenes de todo tipo; además de disponer de un sistema sencillo de anotación. La función del algoritmo dentro del sistema será permitir al usuario anotar las zonas ocluyentes de una escena, como pueden ser mesas o el techo.

- **Obtención de los mapas de confianza de oclusión:** en la sección 2.3 estudiaremos detalladamente el planteamiento teórico propuesto en [2]. En él, se estudia la forma de añadir información de contexto al detector de personas a utilizar, en nuestro caso, el DTDP multiconfiguraciones [4]. En [2] se propone clasificar el contexto según se refiera a la escala o la escena, utilizando para este último caso la imagen anotada en la segmentación.

- **Detección de personas:** en la sección 2.4 se proporciona una visión general de la tarea de detección de personas. Para entender el algoritmo utilizado, veremos con detalle los elementos que componen el detector con múltiples configuraciones que adaptaremos [4]. Este es un detector que toma como base el sistema de detección desarrollado por [3], conocido como DTDP, basado en la detección con diferentes partes del cuerpo, modeladas con el método HOG [5].

2.2. Segmentación

Dentro del procesamiento de imagen y vídeo digital, la segmentación es el proceso de subdividir una imagen en sus partes constituyentes u objetos, con el fin de separar las partes de interés del resto de la imagen. Para realizar el proceso existen múltiples técnicas, difiriendo sus resultados según el tipo de imagen a segmentar. En nuestro

caso, necesitamos una técnica capaz de segmentar imágenes con estructuras complejas. Es por ello que hemos elegido el algoritmo *Watershed* [1], una herramienta utilizada en campos como la biomedicina, donde las imágenes están compuestas por objetos con alto contenido de textura y una amplia variabilidad de formas, tamaños e intensidades. Este algoritmo nos permitirá obtener tanto unos resultados óptimos para imágenes complejas como una sencilla funcionalidad, pues el usuario debe elegir los marcadores para realizar la segmentación.

2.2.1. *Watershed*

El algoritmo *Watershed* [1] es una técnica de segmentación basada en morfología que permite encontrar regiones en una imagen. Esta transformada es fácilmente adaptable a los diferentes tipos de imágenes y es capaz de distinguir objetos sumamente complejos que no pueden ser analizados correctamente mediante algoritmos convencionales. Este método clasifica los píxeles según su proximidad espacial, el gradiente de sus niveles de gris y la homogeneidad de sus texturas.

Con el objeto de segmentar una imagen en niveles de gris, esta se interpreta como una imagen topográfica de un relieve. Mediante el gradiente se obtiene una imagen donde los niveles de los contornos de los objetos a segmentar representan una zona de elevada intensidad de gris y las zonas de baja intensidad de gris darán lugar a las cuencas (*basins*). Como podemos apreciar en la figura 2.2, el algoritmo *Watershed* inunda la imagen gradiente a partir de los mínimos. Tras finalizar la inundación, las elevaciones en los niveles de gris generadas por los contornos permanecerán y darán lugar a la segmentación de la imagen mediante las líneas divisorias correspondientes a una elevada intensidad de gris.

Para evitar la sobresegmentación se recurre a la definición de marcadores unívocos para cada uno de los objetos de interés. El éxito de esta técnica depende de la elección de marcadores unívocos para cada uno de objetos de interés que eviten la sobresegmentación que presentan las imágenes de alto contenido de textura. La obtención automática de marcadores de objetos de gran variabilidad de textura, forma y tamaño requiere un planteamiento complejo y altamente dependiente de cada aplicación en particular. En nuestro caso, los marcadores serán elegidos manualmente, al ser el usuario quien escoge las partes ocluyentes de la escena. La flexibilidad a la hora de elegir los marcadores nos permite disponer de un algoritmo fácilmente adaptable a las características particulares de cada imagen.

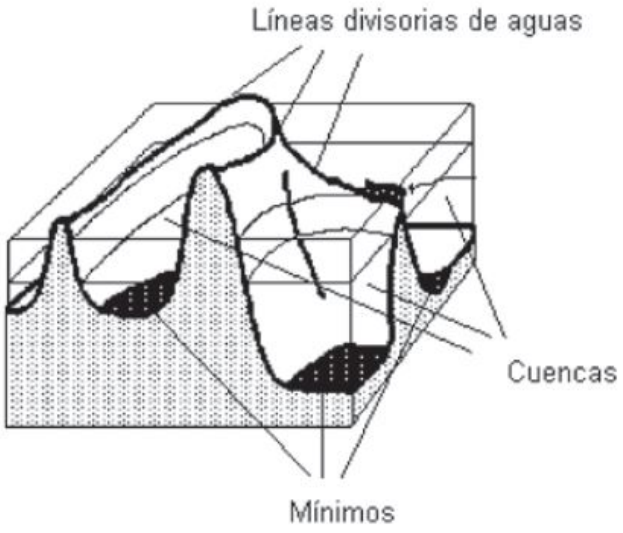


Figura 2.2: Ejemplo del funcionamiento del algoritmo *Watershed* [1], donde la imagen gradiente se inunda. Se pueden apreciar las distintas formas de la imagen dependiendo de la intensidad de gris de cada región. Las líneas divisorias de aguas corresponden a las zonas con mayor intensidad de gris y los mínimos a los valores más bajos, creándose las cuencas alrededor de estos últimos. Imagen extraída de [8].

2.3. Obtención de mapas de confianza de oclusión

Para utilizar la información de contexto en la detección de personas, hay que adaptar el conocimiento de la escena al detector. Como veíamos en la sección 2.2, con el algoritmo *Watershed* podemos obtener una imagen con los objetos ocluyentes anotados. El siguiente paso es saber cómo clasificar esta información y hacerla útil para el detector de personas utilizado, el DTDP multiconfiguraciones [4]. Para ello, utilizamos la aproximación al problema expuesta en [2], implementado en Matlab [7], donde se encuentra el planteamiento principal para detectar personas utilizando información de contexto.

2.3.1. Sistema para detectar personas utilizando contexto de la escena

Para implementar un sistema que tenga en cuenta el contexto de la escena, se debe utilizar un detector de personas del que podamos mejorar su rendimiento. En este caso, se ha adaptado el algoritmo DTDP [3], un detector que considera a cada persona como el filtro principal *root* y P partes del cuerpo (explicado con más detalle en la sección 2.4). Cada parte p está representada por una tupla $\{F_p, D_p, v_p\}$, donde

F_p es el modelo de apariencia, D_p es el modelo de deformación y v_p es la localización óptima de la parte del cuerpo. Detectar personas en una imagen I de dimensiones $M \times N$ implica generar una puntuación s (*score*) para las hipotéticas localizaciones de todas las partes, definidas como $\{l_0, ..., l_p\}$, donde l_p representa las coordenadas (x, y) y la escala a. Para usar el contexto de cada hipótesis, se adapta el DTDP mediante las puntuaciones del contexto para cada parte $\Upsilon(l_p, C_p)$, donde C_p es el conocimiento de la escena para una parte p. La puntuación s para cada hipótesis se calcula según:

$$s(l_0, ..., l_p) = \sum_{p=0}^{P} \Upsilon(l_p, C_p) \left[\langle F_p, \phi(l_p, I) \rangle + \langle D_p, \psi(l_0, l_p) \rangle \right] \tag{2.1}$$

donde $\langle \cdot, \cdot \rangle$ es el producto escalar, $\phi(l_p, I)$ son las características de la imagen I en la posición l_p y $\psi(l_0, l_p)$ es un descriptor de cuatro dimensiones para los desplazamientos entre la posición hipotética l_p y la posición óptima v_p de cada parte respecto a la localización del *root* l_0. Para cada parte, la puntuación del contexto $\Upsilon(l_p, C_p)$ está descompuesta en el contexto local φ^l y el relativo φ^r como:

$$\Upsilon(l_p, C_p) = \varphi^r(\varphi^l(l_p, C_p), \left\{ \varphi^l(l_0, C_0), ..., \varphi^l(l_p, C_p) \right\} \setminus \left\{ \varphi^l(l_p, C_p) \right\}) \tag{2.2}$$

donde el contexto local φ^l se refiere al espacio alrededor de cada parte utilizando el conocimiento de la escena C_p. El contexto relativo φ^r mide la importancia del contexto local para cada parte $\varphi^l(l_p, C_p)$ como comparación con el contexto local de las demás partes $\left\{ \varphi^l(l_0, C_0), ..., \varphi^l(l_p, C_p) \right\} \setminus \left\{ \varphi^l(l_p, C_p) \right\}$. El detector DTDP [3] define una combinación homogénea de las partes, y el contexto en la ecuación 2.1 introduce la heterogeneidad de las mismas, modificando las puntuaciones, lo que produce que no se puedan utilizar los umbrales de la detección original. La función de φ^r es la de calibrar la combinación de partes basada en el contexto para mantener los umbrales originales. Por ejemplo, la importancia relativa de las partes puede ser derivada como la divergencia de *Kullback-Leibler* [9] entre la puntuación de las distribuciones del modelo completo y el modelo por partes [4]. Estimar φ^r no requiere entrenamiento adicional ya que las distribuciones pueden ser compuestas por el entrenamiento original. En este proyecto, vamos a centrarnos en definir el contexto local $\varphi^l(l_p, C_p)$ y el conocimiento C_p.

2.3.2. Contexto local para la detección de personas

El contexto local nos sirve para explorar el espacio a la hora de definir la visibilidad de las partes y para determinar la importancia al combinarlo con el detector

(a) (b) (c)

Figura 2.3: Ejemplos del contexto local referido a la escala. $\varphi^l_{p,a}$ para una imagen de dimensiones 352×288 usando una escala $a = 2$ (dos veces la escala original [3]). El rango de valores varía desde 1 (blanco) a 0 (negro).
(a) Filtro *root*
(b) Filtro de la parte correspondiente a la cabeza
(c) Filtro de la parte correspondiente al hombro izquierdo

DTDP. Dentro del contexto local $\varphi^l(l_p, C_p)$ podemos diferenciar dos tipos de contexto diferentes. El primero de ellos es el contexto relacionado con la escala de detección a. Las partes del modelo pueden estar fuera de las dimensiones de la imagen I en ciertas posiciones y escalas, disminuyendo la eficacia de la detección de personas de este modo. Para obtener el contexto de escala $\varphi^l(l_p, C_p) \equiv \varphi^l_{p,a}(x, y)$ para cada parte en las coordenadas (x, y) y escala a, aplicamos un *kernel* K^a_p sobre una matriz I' todo unos de dimensiones $M \times N$:

$$\varphi^l_{p,a}(x, y) = \sum_{i=x}^{x+M'} \sum_{j=y}^{y+N'} I'(i, j) \cdot K^a_p(i + d_x - x, j + d_y - y) \tag{2.3}$$

donde (i, j) son las coordenadas de los píxeles, (d_x, d_y) son los desplazamientos de las partes respecto al centro del filtro *root* y K^a_p es una matriz $M' \times N'$ todo unos, donde el tamaño es el de la parte del modelo de apariencia F_p reescalado por el factor a. $\varphi^l_{p,a}(x, y)$ estima la similitud para detectar la parte p en la imagen I en la escala a y la posición (x, y). Por consiguiente, el contexto local referido a la escala para cada una de las partes es definido como el tamaño de los *kernels* a la escala correspondiente, $C_p = \{M' \times N'\}_{a=0...A}$. La figura 2.3 representa ejemplos del contexto local referido a la escala.

Por otro lado, también estimamos el contexto local del conocimiento de los descriptores del espacio como los objetos estáticos de la escena [10]. Descriptores que están combinados con las restricciones espaciales de las reglas semánticas en un sistema ontológico [10]. Por ejemplo, algunas detecciones pueden ser ignoradas como encontrar piernas en el techo de una escena, cabezas en el suelo o partes del cuerpo

(a) (b) (c)

Figura 2.4: Ejemplos del contexto local referido a la escena. $\varphi^l_{p,s}$ para la *dataset* EDds (*Event Detection dataset*) [11], usando dos veces la escala original [3]. Para los mapas de confianza, el rango de valores varía desde 1 (blanco) a 0 (negro).
(a) Filtro *root*
(b) Filtro de la parte correspondiente a la cabeza
(c) Anotación de los elementos estáticos (cada uno en un color)

ocluidas por una mesa. Si asumimos que el contexto no varía con el tiempo, se puede aplicar para la vídeo-seguridad con cámaras estáticas. Para obtener el contexto referido a la escena $\varphi^l_s(l_p, C_p) \equiv \varphi^l_{p,s}(x, y)$ para cada parte p y escala a, aplicamos la similitud del *kernel* previamente definido K^a_p a la ecuación 2.3:

$$\varphi^l_{p,s}(x,y) = \min_{\forall_0 \in \mathcal{O}_p} \left(\sum_{i=x}^{x+M'} \sum_{j=y}^{y+N'} \mathcal{M}_0(i,j) \cdot K^a_p(i + d_x - x, j + d_y - y) \right) \tag{2.4}$$

donde $\mathcal{O}_p$ es el grupo de elementos estáticos o que pueden ocluir (p. ej. mesas) o evitarse (p. ej. techos) la detección de una determinada parte p; $|\cdot|$ es el conjunto de cardinalidad y $\mathcal{M}_0$ es una matriz $M \times N$ binaria que indica la posición del objeto o que se puede obtener mediante herramientas de anotación [10]. En nuestro caso, se utilizará la imagen con las zonas oclusivas anotadas resultante de la etapa de segmentación comentada en la sección 2.2. Las reglas semánticas están representadas por conjuntos $\mathcal{O}_p$, relacionando cada parte con los objetos estáticos que afectan su visibilidad y, además, determinan el conocimiento de cada parte como $C_p = \{\mathcal{O}_p, \mathcal{M}_0\}$. La figura 2.4 representa ejemplos del contexto local referido a la escena.

2.4. Detección de personas

La detección de personas es uno de los mayores retos del procesamiento de imagen y vídeo digital, ya que las personas a detectar tienen una gran variabilidad no sólo debida a cambios en la iluminación, el color o ángulos de visión sino también debida a la propia variación de la forma. Se ha de tener en cuenta que, en muchas ocasiones algunas partes de la personas no aparecen en el vídeo. Un problema habitual es cuando las extremidades inferiores de la persona se encuentran fuera del vídeo o

están ocluidas por objetos estáticos. Para mejorar este problema, se van a emplear los mapas de confianza vistos en la sección 2.3.

Para utilizar la información de contexto consistente en evitar buscar partes de la persona en la escena, debemos adaptar un detector de personas con diferentes configuraciones de partes de las personas. En nuestro caso, el detector escogido es el DTDP multiconfiguraciones [4]. Su características principales son dividir el modelo de persona en partes y tener una componente deformable que puede ser caracterizada por la conexión entre pares de partes cercanas. Estas condiciones serán muy favorables para nuestro objetivo, ya que en el DTDP original [3] las partes son definidas como un conjunto.

Antes de estudiar el algoritmo DTDP multiconfiguraciones en la sección 2.4.3, explicaremos brevemente el estado del arte de los dos algoritmos que forman la base de este detector, HOG en la sección 2.4.1 y DTDP en la sección 2.4.2.

2.4.1. *Histogram of Oriented Gradient*

En la detección de personas, las técnicas basadas en la apariencia definen descriptores, imágenes de características, a partir de las cuales se clasifican las ROI (*Region of Interest*). Estas regiones se aplican si pueden contener personas y se descartan aquellas que no las contienen. Un ejemplo de este tipo de algoritmo es el propuesto por [5], denominado HOG (*Histograms of Oriented Gradient*). Este método consiste en la evaluación de histogramas locales normalizados de las orientaciones de los gradientes de una imagen. Se implementa dividiendo la imagen en pequeñas regiones espaciales (*cells*), acumulando en cada una de ellas un histograma, de una dimensión, de las direcciones del gradiente o de las orientaciones del borde sobre los píxeles que se encuentran dentro de cada *cell*. La combinación de estos histogramas forma la representación que vemos en la figura 2.5.

Para conseguir una mejor invariancia frente a la iluminación, las sombras, etc., se recomienda normalizar el contraste de las zonas locales antes de operar con ellas. Esto se puede conseguir acumulando la energía de los histogramas locales sobre una región mayor (*blocks*) y usar este resultado para normalizar todos los *cells* que están contenidos en dicho *block*. A estos bloques normalizados los llamaremos descriptores HOG *(Histogram of Oriented Gradient*). Estos descriptores se obtendrán de una ventana de detección dividida en una cuadrícula, con cierto grado de solape entre cuadros y usando SVM (*Support Vector Machine*) lineal se hará la clasificación como personas o no.

HOG captura los bordes o estructuras de gradientes más característicos de una zona local y lo hace con un cierto grado de invariancia a transformaciones, es decir,

Figura 2.5: Ejemplo de descriptores HOG. Imagen extraída de [5].
(a) Promedio de los gradientes en la imagen de entrenamiento
(b) Imagen de entrenamiento
(c) Descriptor HOG

las traslaciones o rotaciones apenas afectan al resultado. Estas características hacen que el método presente buenos resultados, siendo su principal desventaja el elevado coste computacional.

2.4.2. *Discriminatively Trained Deformable Part-based*

DTDP (*Discriminatively Trained Deformable Part-based model*) [3] es un sistema de detección basado en mezclas de modelos de partes deformables multiescala definido por un filtro *root* que, junto con el resto de partes deformables del cuerpo, están modelados por HOG, como en primer lugar propuso [5]. El detector propone N partes del cuerpo posicionadas alrededor del filtro *root*, filtro principal que define la persona en su totalidad ($n = 0$), como se muestra en la figura 2.6. Estos filtros correspondientes a las diferentes partes del cuerpo están definidos a doble de resolución y cubren pequeñas zonas del cuerpo, tal y como puede verse en la figura 2.7, estos filtros de pequeñas partes del cuerpo identifican cada zona en mayor detalle.

Cada parte del modelo, incluyendo el *root*, ($n = 0, ..., N$) está definido por tres variables $F_n, v_{n,0}, d_n$; donde Fn es la respuesta al filtro HOG para la parte n. Por su parte, $v_{n,0}$ es un vector de dos dimensiones que contiene la posición relativa de la parte n, con respecto al *root*, y d_n es un vector de cuatro dimensiones que define los coeficientes de una función cuadrática que define la deformación de cada parte n. Por tanto, la puntuación de cada parte, se obtiene de las siguientes ecuaciones 2.5, 2.6 y 2.7.

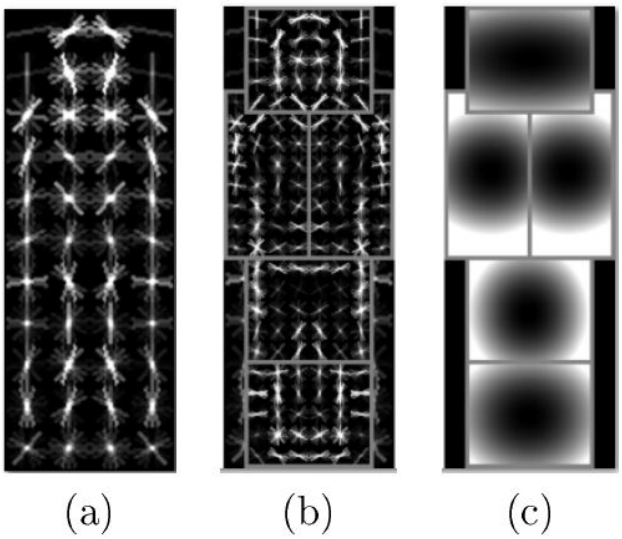


Figura 2.6: Ejemplo de filtro *root* y filtros de partes del cuerpo en Latent SVM. Imagen extraída de [3].
(a) Filtro *root*
(b) Filtros de cada parte del cuerpo a doble de resolución
(c) Modelo espacial de la localización de cada filtro respecto al filtro *root*

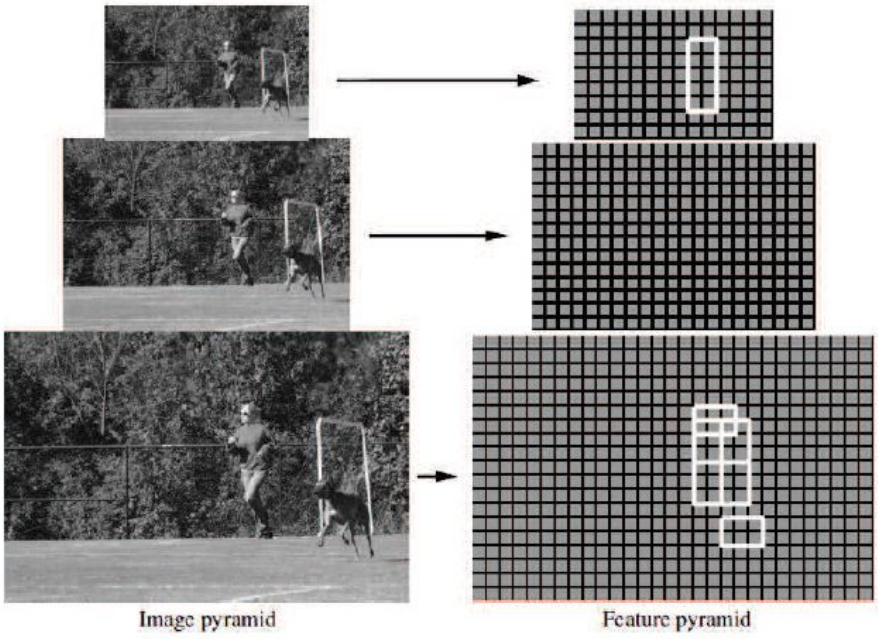


Figura 2.7: Pirámide de características. Los filtros de partes del cuerpo se encuentran en el nivel de la pirámide con doble resolución respecto al filtro *root*. Imagen extraída de [3].

$$BP_n\,(x,y,s) = F_n(x,y,s) - \langle d_n, \Phi(dx_n, dy_n)\rangle \tag{2.5}$$

$$(dx_n, dy_n) = (x_n, y_n) - (2(x_0, y_0) + v_n, 0) \tag{2.6}$$

$$\phi(dx, dy) = (dx, dy, dx^2, dy^2) \tag{2.7}$$

donde en la ecuación 2.5, BP_n representa la puntuación de un píxel en la posición (x, y) para la parte del cuerpo n donde s identifica la escala $(s = 1, ..., S)$, la ecuación 2.6 representa el desplazamiento de la parte n con respecto al *root* y la ecuación 2.7 la distribución de la deformación espacial de la parte n.

El resultado final de la detección viene dado por $C(x, y, s)$, como la suma del *root* y todas las partes en cada píxel y escala. Una puntuación alta determinará una detección, aunque como método adicional para evitar falsos positivos (detección de un objeto en un lugar de la imagen donde no los hay) se eliminan aquellas detecciones que tengan un solape superior al 50 %. El umbral escogido, por encima del cual se detectará una persona, depende en [3] directamente del número total de partes del cuerpo detectadas.

$$C(x,y,s) = \sum_{n=0}^{N} BP_n(x,y,s) \tag{2.8}$$

A modo de ejemplo en la figura 2.8 podemos ver un esquema de cómo funciona DTDP. Se crean dos mapas de características, uno de ellos se comparará con el filtro *root*; mientras que el otro, al doble de resolución, se comparará con los filtros por partes. De la combinación de todas las puntuaciones obtenemos el resultado final y la detección.

La principal dificultad de este tipo de sistemas se encuentra en que son muy complicados de entrenar debido a que en muchas ocasiones se necesita información latente, las imágenes de entrenamiento sólo disponen de la información de un rectángulo alrededor del objeto, se desconoce, por tanto, dónde están situadas las diferentes partes del cuerpo. Con un etiquetado se podría aportar dicha información, aunque supondría un importante gasto de tiempo.

2.4.3. Múltiples configuraciones de partes del cuerpo

Partiendo de la base de DTDP explicada en la sección 2.4.2, en [4] se propone un método para mejorar este algoritmo aplicado a la detección de personas en grupos permitiendo que el algoritmo no base su decisión únicamente en una sola configuración

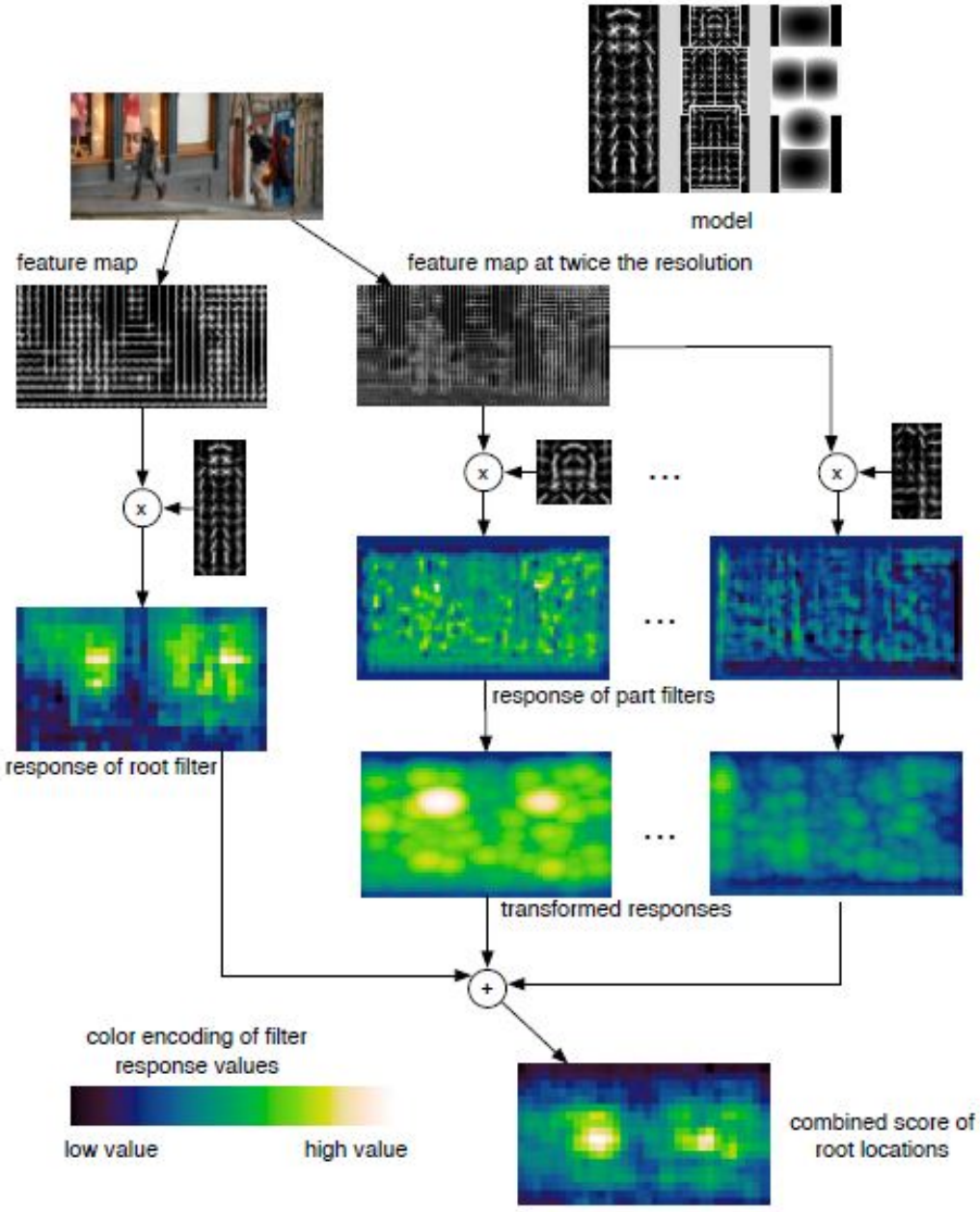


Figura 2.8: Esquema general del detector DTDP. Imagen extraída de [3].

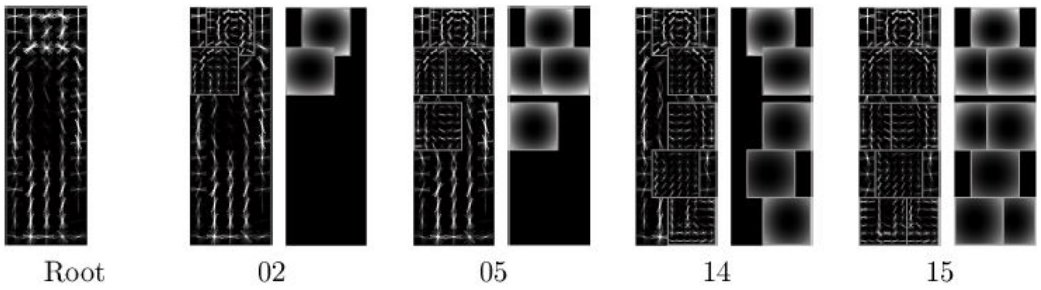


Figura 2.9: Ejemplos de las configuraciones. La primera columna se corresponde con el filtro *root*. El resto de columnas corresponde a las configuraciones diseñadas, a cada configuración se le ha dado un código numérico. Las nueve partes del cuerpo serían: cabeza, hombro derecho, hombro izquierdo, tronco derecho, tronco izquierdo, entrepierna, pierna derecha, pierna izquierda y el *root*.

de las partes del cuerpo detectadas. Para ello propone el uso de diferentes configuraciones de partes del cuerpo de personas, $t(t = 1, ..., T)$ donde $1 \leq T \leq 2^N$ y cada configuración t se compone de un subconjunto de las N partes del modelo original [3]. Este algoritmo diseña treinta configuraciones diferentes ($T = 30$). De las cuales, quince configuraciones se diseñan combinando diferentes partes del cuerpo, y otras quince configuraciones, idénticas a las anteriores, que contienen, además, el filtro *root*. Todas las configuraciones van añadiendo progresivamente partes del cuerpo desde la cabeza ($t = 1$), hasta todas las ocho partes del cuerpo definidas ($t = 15$). En la figura 2.9 se puede ver el ejemplo de alguna de estas configuraciones.

Asimismo se diferencia del algoritmo original (ecuación 2.8 en la sección 2.4.2), pues el resultado para cada configuración es definido por las ecuaciones 2.9 y 2.10 donde α^t es un vector binario para cada configuración.

$$C_t(x, y, s) = \sum_{n=0}^{N} \alpha_n^t \cdot BP_n(x, y, s) \tag{2.9}$$

$$\alpha_n^t = \begin{cases} 1, & n \subset t \\ 0, & otherwise \end{cases} \tag{2.10}$$

Las configuraciones diseñadas en este algoritmo se diseñaron pensando en la aplicación específica de detección de personas en grupos por lo que se ha tenido en cuenta el tipo de oclusión que se da en estos casos. En muchos casos, las oclusiones son generadas por otras personas e imposibilitan ver las extremidades inferiores, por eso se diseñan configuraciones donde sólo se tienen en cuenta partes potencialmente visibles del cuerpo. La detección de personas en grupos también es un problema en el que se

debe solucionar las oclusiones, por lo que es perfecto para adaptar el detector al contexto. Como hemos visto en la sección 2.3, con la adicción de los mapas de confianza el detector DTDP multiconfiguraciones mejora su eficiencia utilizando información de contexto, siendo su variable DTDP-Context.

Capítulo 3

Diseño y desarrollo

Tras estudiar en detalle los algoritmos existentes en el capítulo anterior, describiremos detalladamente el desarrollo del sistema a implementar. Como explicábamos en el estado del arte, el diseño del sistema se divide en tres etapas: segmentación, obtención de mapas de confianza de oclusión y detección de personas. En este capítulo desarrollaremos el diseño y la implementación de cada una de las etapas:

- **Creación de un método para anotar zonas oclusivas:** tomando como base el algoritmo en C++ de segmentación morfológica *Watershed* [1], ampliaremos su funcionalidad con el objetivo de poder anotar las zonas oclusivas de una imagen de manera sencilla.
- **Implementación en C++ del algoritmo para obtener los mapas de confianza de oclusión:** se ha implementado en C++ el algoritmo de obtención de mapas de confianza de oclusión existente en Matlab [7]. Esta se ha realizado siguiendo las especificaciones del sistema estudiado en la sección 3.2 del estado del arte.
- **Modificación del detector DTDP multiconfiguraciones para incorporar información de contexto:** modificación del algoritmo de detección de personas DTDP, disponible en las librerías de OpenCV [6], para añadir los mapas de confianza de oclusión.

3.1. Segmentación

El primer paso de nuestro sistema consiste en anotar las zonas oclusivas de la escena. Por ejemplo, en una secuencia con una mesa y un techo donde sabemos por lógica que no deberían aparecer personas, debemos segmentar la imagen en regiones

con la mayor precisión posible y anotar la zonas donde descartaremos todas aquellas detecciones erróneas de personas. Como explicábamos en la sección 2.2.1 del estado del arte, el algoritmo de segmentación elegido es *Watershed* [1]. Las principales características del mismo son los buenos resultados obtenidos para segmentar imágenes complejas y su funcionalidad simple. Estas prestaciones se adecuan al objetivo de esta etapa del sistema, teniendo que crear un algoritmo que nos permita anotar las zonas oclusivas de forma sencilla e intuitiva.

3.1.1. Anotación de zonas oclusivas

Para la creación del algoritmo de anotación de zonas oclusivas, se ha tomado como base el método *Watershed* [1], disponible en las librerías de OpenCV [6]. Este algoritmo nos proporcionará la segmentación de la imagen, añadiéndole la funcionalidad de la anotación de las zonas oclusivas. Para describir el diseño de esta etapa, vamos a explicar los pasos esenciales del algoritmo implementado tras añadir su nueva funcionalidad:

1. **Selección de la imagen a anotar:** este trabajo tiene como objetivo mejorar las detecciones de personas en vídeos de escenas estáticas. En este tipo de secuencias, los objetos estáticos a anotar no cambian su localización, es por ello que la imagen escogida es el primer *frame* de la secuencia.

2. **Definición de los marcadores unívocos:** el método de segmentación *Watershed* [1] basa su éxito en la elección de marcadores unívocos para cada uno de los objetos de interés. El usuario debe seleccionar tantos marcadores como zonas de la imagen quiera obtener, ya que como veíamos en la sección 2.2.1 del estado del arte, desde ellos se inundará el relieve de los niveles de gris. Los marcadores se definen como trazados blancos marcados sobre la imagen. En la imagen a segmentar se pueden generar múltiples trazados, si estos son conexos significa que forman un único marcador y definen una misma región y si no están conectados, indican que son regiones separadas.

   El código utilizado tiene una sencilla funcionalidad: el usuario marca las zonas de forma manual, pudiendo tanto iterar añadiendo nuevos marcadores para mejorar la precisión como volver a empezar desde el principio el proceso de anotación si ha ocurrido algún error. Como podemos apreciar en la figura 3.1, normalmente no se van a marcar una gran cantidad de zonas. Seleccionando sin mucha precisión los bordes de los objetos estáticos y el fondo podemos conseguir una segmentación con resultados casi óptimos.

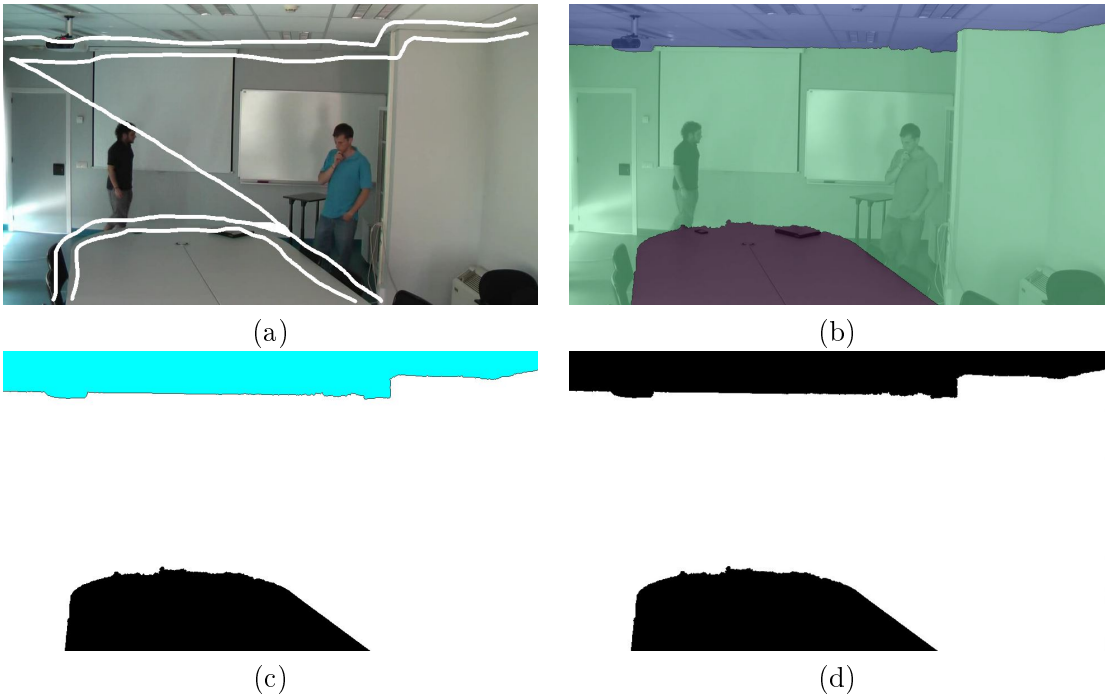


Figura 3.1: Ejemplo de anotación de zonas oclusivas, donde la imagen final se ha obtenido a través de los siguientes pasos: selección del primer *frame*, definición de los marcadores unívocos para cada zona, segmentación y método para anotación de las zonas oclusivas. Imagen extraída de la *dataset* EDds (*Event Detection dataset*) [11].
(a) Marcadores, en blanco, sobre el primer *frame* de la secuencia: techo, fondo y mesa
(b) Imagen segmentada
(c) Selección para la anotación de zonas oclusivas. En azul la zona a etiquetar entre blanco (no oclusivo) y negro (oclusivo)
(d) Imagen final con el techo y la mesa marcados como oclusivos en color negro

3. **Segmentación:** para la segmentación se utiliza la función *Watershed* presente en las librerías de OpenCV [6]. Los argumentos de esta función son la imagen a segmentar y los marcadores unívocos, dependiendo el resultado de la precisión de estos últimos.

4. **Anotación de las zonas oclusivas:** una vez segmentada la imagen con éxito, debemos anotar las zonas oclusivas. Para ello hemos creado un método que permite a los usuarios anotar las zonas oclusivas de la escena de forma sencilla e intuitiva. Tras finalizar la segmentación, se irán marcando las zonas segmentadas en azul claro, como puede verse en la figura 3.1 (c). Para cada zona marcada en azul, el usuario deberá elegir etiquetarla como zona oclusiva o no oclusiva, dándole el color negro y blanco respectivamente. Al igual que el método de elección de marcadores, se podrá reiniciar la anotación para subsanar cualquier error durante el proceso de anotación.

En la figura 3.1 podemos observar un ejemplo de los distintos pasos que conforman esta etapa, siendo el resultado la imagen con las zonas oclusivas anotadas.

3.2. Obtención de mapas de confianza de oclusión

Tras obtener una imagen con las zonas oclusivas anotadas, debemos adaptar esta información a las características del detector. En la sección 2.3 del estado del arte explicábamos la aproximación al problema expuesta en [2], donde se desarrolla un método para adecuar el conocimiento de la escena al detector elegido, el DTDP multiconfiguraciones [4]. Para obtener los mapas de confianza de oclusión determinamos dos pasos: el estudio del detector DTDP multiconfiguraciones y su implementación, y el método utilizado para obtener los mapas.

3.2.1. Cálculo de mapas de confianza de oclusión

Para implementar un algoritmo con el que obtener los mapas de confianza, debemos estudiar la manera en la que el detector determina los candidatos a persona. En nuestro caso, el objetivo es poder dar más relevancia a ciertas configuraciones de partes del cuerpo según la naturaleza de la escena. En la sección 2.4.3 del estado del arte, explicábamos cómo el detector DTDP multiconfiguraciones definía las diferentes partes del cuerpo. Este detector propone N partes del cuerpo posicionadas alrededor del filtro de la persona completa, denominado *root*. Los filtros correspondientes a las partes del cuerpo se definen a doble resolución, teniendo lugar la búsqueda de dichos filtros en niveles de escala distintos. El detector DTDP multiconfiguraciones

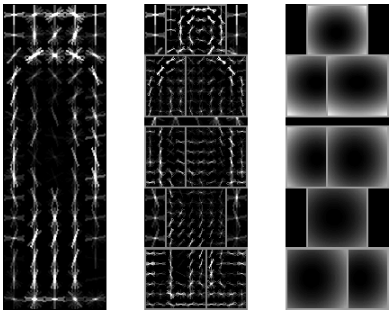


Figura 3.2: Modelo de persona utilizado. La primera columna se corresponde con el filtro *root*; la segunda columna, con los filtros de cada parte del cuerpo al doble de resolución, y la tercera con el modelo espacial de localización de cada filtro con respecto al *root*. Las nueve partes del cuerpo serían: cabeza, hombro derecho, hombro izquierdo, tronco derecho, tronco izquierdo, entrepierna, pierna derecha, pierna izquierda y *root*, persona completa. Imagen extraída de [3].

crea una pirámide de características donde calcula mapas de características de la imagen a diferentes resoluciones, generando múltiples niveles en función del tamaño de la imagen.

La utilización del detector DTDP multiconfiguraciones exige la elección de un modelo de persona. En el estado del arte, existen distintos modelos con diferente número de filtros de partes del cuerpo y múltiples configuraciones. En este trabajo, se ha elegido un modelo de persona con nueve filtros: *root*, cabeza, hombro derecho, hombro izquierdo, tronco derecho, tronco izquierdo, entrepierna, pierna derecha y pierna izquierda. Como podemos observar en la figura 3.2, la disposición de los nueve filtros nos permite obtener un modelo de persona altamente deformable y poder adaptarse a diversas oclusiones. A la hora de obtener los mapas de confianza de oclusión, se variará la resolución de los distintos filtros para generar la pirámide de mapas de confianza de oclusión con diferentes niveles de escala.

El algoritmo para obtener los mapas de confianza de oclusión tiene como objetivo crear una serie de imágenes donde el rango varía desde 1 (blanco) a 0 (negro). Estos colores determinan las zonas donde el detector puede eliminar los candidatos a persona o validar los mismos. En el detector implementado que explicaremos en la sección 3.3, el planteamiento es tomar como útil una detección cuando los valores de los mapas para la posición central del *blob*[1] coincidan con la configuración de la detección. Para calcular los mapas de confianza según la resolución y la posición de los filtros dentro del modelo de persona, realizamos una convolución de imágenes.

[1] *Blob* (*Binary Large Object*) es un conjunto de píxeles similares entre ellos y a su vez contrastados con el resto de los píxeles presentes en la imagen. Se define por un rectángulo con coordenadas (x, y) en uno de sus vértices, ancho y alto.

Figura 3.3: Ejemplo de mapa de confianza de oclusión. Resultado de la convolución entre la imagen con una mesa y el techo etiquetados como oclusivos y el filtro *root*.

En este algoritmo calcularemos los mapas de confianza con una convolución entre la imagen con las zonas oclusivas anotadas y con los diferentes filtros con las resoluciones definidas. La operación de convolución puede expresarse como:

$$\theta[n,m] = \psi[n.m] * h[n,m] = \sum_{k=-a}^{a} \sum_{l=-b}^{b} \psi[k,l] \cdot h[n-k, m-l] \tag{3.1}$$

donde $\theta[n,m]$ se obtiene centrando $h[-n,-m]$ sobre cada píxel $\psi[n,m]$ y sumando los productos de cada coeficiente de h por cada píxel homólogo de la vecindad. En nuestro caso, ψ se corresponde con la imagen con las zonas oclusivas anotadas y el *kernel* h con los filtros a las distintas resoluciones, compuesto por las partes del cuerpo correspondientes a su configuración y su posición relativa respecto al *root*. El resultado de la ecuación 3.1 nos devuelve una imagen, que tras normalizar, presenta valores entre 0 y 1. Umbralizando con un valor de umbral de 0.5, valor discriminante entre zonas donde se puede ver la mitad de las partes de la configuración correspondiente, obtenemos un mapa donde se diferencian las zonas blancas (valor 1), donde validaremos las detecciones, y las negras (valor 0), donde las eliminaremos. En la figura 3.3 se puede apreciar un ejemplo de mapa de confianza de oclusión sin umbralizar para el filtro *root*, para poder entender cuál es el resultado de la convolución entre la imagen con las zonas oclusivas anotadas y el filtro.

Una vez estudiado el algoritmo de obtención de mapas de confianza de oclusión, debemos relacionar los mapas con los niveles de escala utilizados en la detección por el algoritmo DTDP multiconfiguraciones. Dependiendo del nivel de escala, la resolución de cada filtro cambiará, obteniendo mapas de confianza de oclusión con predominancia del blanco, zonas donde detectar, o del negro, donde eliminar los candidatos a personas.

3.2.2. Adaptación al detector DTDP multiconfiguraciones

Para adaptar los mapas de confianza al detector DTDP multiconfiguraciones, la idea fundamental es replicar la función del detector que crea la pirámide de características. La función del algoritmo genera diferentes niveles de escalas, donde se buscarán candidatos a personas de diferentes resoluciones. Como comentábamos en la sección 2.4.3 del estado del arte, el detector DTDP multiconfiguraciones busca el filtro *root* en un nivel de escala y los filtros de las partes del cuerpo a doble resolución. Conociendo las propiedades del método en el detector DTDP multiconfiguraciones, se ha creado una función en C++ denominada *confidencepyramid*. Esta tiene como argumento de entrada la imagen con las zonas de la escena anotadas, en negro las zonas oclusivas y en blanco el fondo; retornando los mapas de confianza de oclusión para todos los niveles de escala de detección.

El desarrollo de la adaptación consiste en reescalar los nueve filtros de nuestro modelo de persona y obtener la convolución con la imagen con las zonas oclusivas anotadas. Este método nos permite generar una pirámide de mapas de confianza de oclusión, desde niveles de escala con valores bajos a niveles altos. El nivel de escala define la resolución de los filtros, lo que implica que a niveles altos la persona a detectar tendrá mayor tamaño. Según el nivel de escala tendremos mapas de confianza de oclusión en los que será más difícil detectar personas que en otros.

La información de contexto anotada en la primera etapa del sistema se adapta al detector por medio de los mapas de confianza. La obtención de los mismos nos permite entender la manera de detectar del DTDP multiconfiguraciones. Como podemos observar en los ejemplos presentes en la figura 3.4, dependiendo a qué parte del cuerpo corresponde el filtro, habrá zonas de la imagen en las detecciones no sean válidas. En la figura 3.4 (e), podemos apreciar que al ser un mapa correspondiente al filtro de la cabeza a un nivel de escala alto (persona de gran tamaño), lo normal es que no se encuentre la cabeza de la persona en la escena. Esto se debe a que para una persona que ocupa gran parte de la resolución, la cabeza se sale por el borde superior, ya que el mapa se crea en relación al centro de una hipotética persona donde la cabeza quedaría fuera de la resolución de la escena. Esta idea se traduce en un mapa de confianza de oclusión con una división entre negro y blanco en la mitad del mismo. El mismo razonamiento sirve para las partes del cuerpo restantes, pues al aumentar la resolución de los filtros se acentúan las zonas negras de los mapas, donde eliminaremos los candidatos de personas.

En cuanto al incremento de tiempo de ejecución que implica la obtención de los mapas de confianza, señalar que los mapas sólo se obtienen una vez por cada vídeo a procesar. Además, debido a lo exhaustivo del detector DTDP multiconfiguraciones [4],

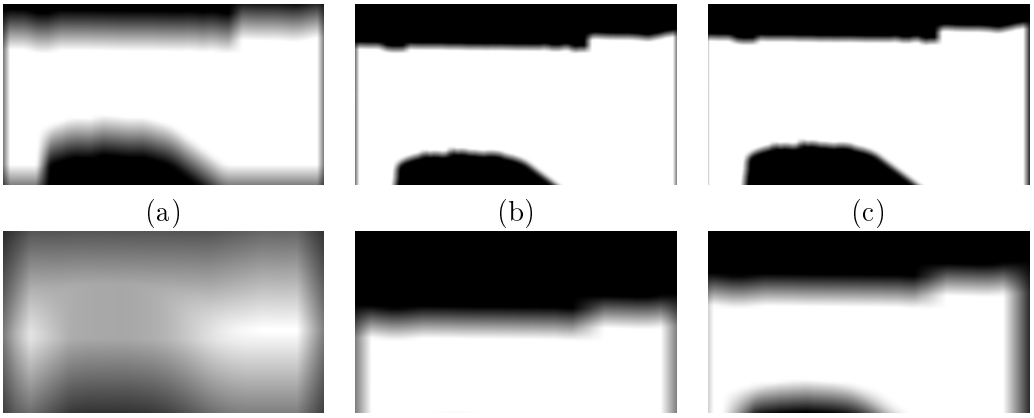


Figura 3.4: Ejemplos de mapas de confianza de oclusión a distintos niveles escala. Mapas resultantes del algoritmo de obtención de mapas de confianza para filtros con dos resoluciones distintas. Para los mapas de confianza antes de umbralizar, el rango de valores varía desde 1 (blanco) a 0 (negro).
(a), (b) y (c) Mapas de los filtros *root,* cabeza y hombro derecho para el nivel uno de la pirámide
(d), (e) y (f) Mapas de los filtros *root,* cabeza y hombro derecho para el nivel treinta de la pirámide

este algoritmo no funciona en tiempo real. Esto implica que el tiempo suplementario al obtener los mapas está justificado en la mejora de los resultados obtenidos.

3.3. Detección de personas

En la tercera etapa del desarrollo de nuestro sistema, modificaremos el detector DTDP para incorporar información de contexto. Tras haber anotado las zonas oclusivas y obtenido los mapas de confianza, debemos integrar el conocimiento de la escena en el detector de forma eficiente. Para conseguirlo, en primer lugar estudiaremos el funcionamiento del detector DTDP [3], disponible en las librerías de OpenCV [6], y la adaptación realizada para tener múltiples configuraciones, es decir, el detector DTDP multiconfiguraciones [4]. Una vez analizado el funcionamiento del detector, incorporaremos los mapas de confianza en busca de mejorar la tarea de detección con la información de contexto extraída previamente.

3.3.1. Latent SVM y DTDP multiconfiguraciones

Para poder describir la incorporación de información de contexto a la detección de personas, vamos a explicar la versión en C++ del detector DTDP multiconfiguraciones

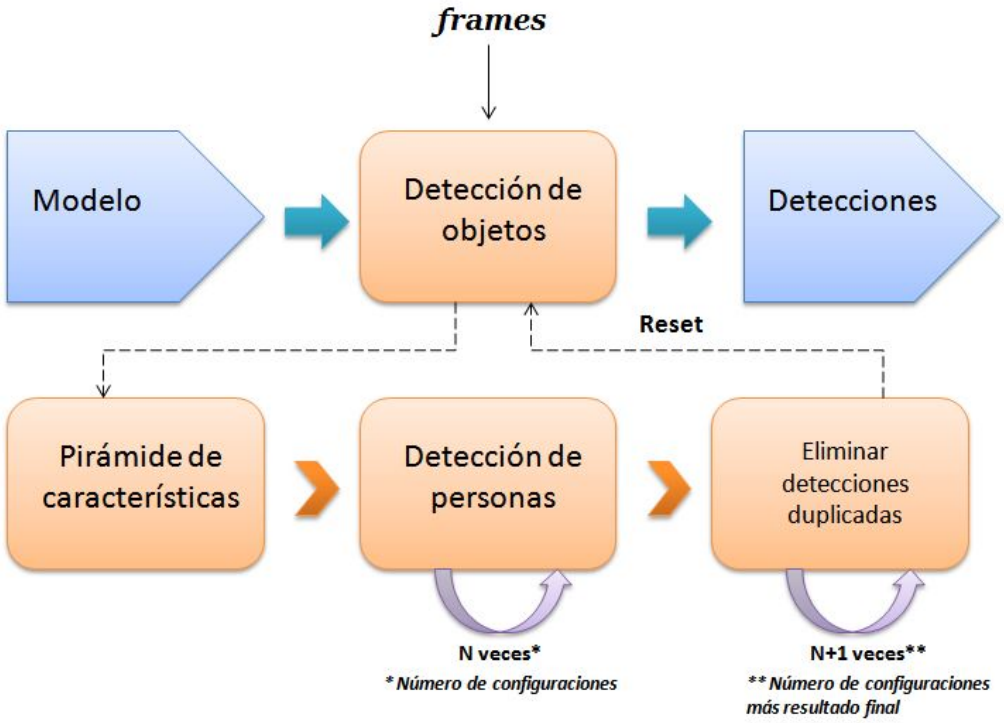


Figura 3.5: Esquema de la implementación del detector DTDP multiconfiguraciones. La diferencia entre el esquema del detector DTDP multiconfiguraciones respecto al del algoritmo Latent SVM es la inclusión de las múltiples configuraciones, representada en el esquema con los elementos de color morado.

implementado en [4]. Este algoritmo toma como base el detector DTDP [3], disponible en las librerías de OpenCV [6] bajo el nombre de Latent SVM. Por este motivo vamos a describir el funcionamiento del DTDP multiconfiguraciones antes de desarrollar cómo hemos incorporado la información de contexto.

En la figura 3.5 podemos ver de forma simplificada el esquema que sigue DTDP multiconfiguraciones y se encuentra implementado de la siguiente manera:

1. **Modelo de persona:** Primero se carga el modelo de persona. Al tener distintas configuraciones, el fichero que contiene el modelo deberá definir la disposición de cada una de las configuraciones, así como el umbral utilizado en la detección. Para leer el fichero de configuración se emplea el método *parse_ configurations*, cuya función es leer el fichero y calcular los niveles de confianza para cada configuración. Como explicamos en la sección 2.4 del estado de arte, el umbral dependerá del número y tipo de partes del cuerpo que compongan cada configuración.

2. **Detección de personas:** Una vez obtenido el modelo, se analizarán uno por uno los *frames* del vídeo. Para ello se hará uso de la función *cvLatentSvmDetectObjects*, que encontrará *blobs* en la imagen que tengan niveles de confianza adecuados como para ser considerados personas. Para ello llamamos a la función de la siguiente forma:

   *CvSeq** cvLatentSvmDetectObjects(IplImage* image,CvLatentSvmDetector* detector, CvMemStorage* storage, float *overlap_ threshold, int numThreads, float *score_ threshold, int N_ configurations, int **configuration)*

   Donde:

   - ***Image*** corresponde al frame que se está procesando.
   - ***Detector*** contiene el modelo de personas entrenado.
   - ***Storage*** especifica un almacenamiento en memoria donde se guardará la secuencia resultante de rectángulos encontrados y puntuaciones correspondientes.
   - ***Overlap_ threshold*** contiene el umbral permitido de solape entre detecciones.
   - ***NumThreads*** especifica el número de hilos permitido, para aquellas ejecuciones que permitan una ejecución con varios hilos.
   - ***Score_ threshold*** define los umbrales de cada configuración.
   - ***N_ configurations*** especifica el número de configuraciones.
   - ***Configuration*** contiene los datos referentes a las configuraciones.

   Dentro de *cvLatentSvmDetectObjects,* la detección se realiza en tres pasos principales:

   - En primer lugar, se crea una pirámide de características usando *createFeaturePyramidWithBorder*. Esta función calcula mapas de características de la imagen a diferentes resoluciones, generando múltiples niveles de escala en función del tamaño de la imagen. Como hemos comentado en la sección 3.2, el algoritmo implementado para obtener los mapas de confianza replica su funcionamiento.
   - En segundo lugar, se obtienen las detecciones con la función *searchObjectThresholdSomeComponents*, que recibe como argumentos la pirámide de características, los filtros HOG [5] y el umbral de detección. Esta función se encarga de comparar el mapa de características con las diferentes

configuraciones, es decir, detectar para cada mapa de características a diferentes resoluciones los diferentes filtros que componen cada configuración. Todos aquellos objetos que obtengan una puntuación mayor al umbral de detección serán clasificados como personas, devolviendo las coordenadas y puntuaciones de las detecciones obtenidas con las diferentes configuraciones.

- Por último, es necesario eliminar posibles detecciones duplicadas mediante *nonMaximumSuppression*. Esta función elimina aquellas detecciones que superen un cierto nivel de solape, evitando obtener falsos positivos.

3. **Representación de las detecciones:** Finalmente, dibujamos las detecciones resultantes haciendo uso de la función *cvRectangle*, que dibujará un rectángulo por cada detección.

Debido a la naturaleza exhaustiva del detector, este algoritmo no se puede analizar en tiempo real. Al tener que anotar las zonas oclusivas y obtener los mapas, el detector está pensado para ejecutar vídeos pertenecientes a una base de datos; aunque sería posible ejecutar sobre una cámara procesando con un *frame rate* muy bajo.

Una vez analizado el funcionamiento del algoritmo DTDP multiconfiguraciones, explicaremos la implementación de la incorporación de información de contexto. Este es el último paso del desarrollo del sistema, debiendo integrar al detector el conocimiento de la escena obtenido en las dos etapas anteriores.

3.3.2. Inclusión de información de contexto

Con el fin de integrar información de contexto en el detector DTDP multiconfiguraciones, vamos a modificar el esquema de la implementación del detector visto en la figura 3.5. En la sección 3.3.1 estudiábamos el funcionamiento en C++ del detector, compuesto por tres pasos principales: la creación de una pirámide de características, la obtención de las detecciones y la eliminación de las detecciones duplicadas. Para añadir los mapas de confianza de oclusión, vamos a implementar un cuarto paso.

El diseño para incluir la información de contexto es eliminar las posibles detecciones cuya configuración no se pueda encontrar debido a elementos oclusivos o la propia resolución de la imagen. Por este motivo, tras el método para eliminar las posibles detecciones duplicadas mediante el solape, filtraremos las detecciones restantes con los mapas de confianza. En la figura 3.6 podemos apreciar el esquema simplificado del nuevo detector implementado.

Entre las detecciones obtenidas tras la eliminación de las detecciones duplicadas y la representación con rectángulos de las detecciones resultantes, hemos añadido

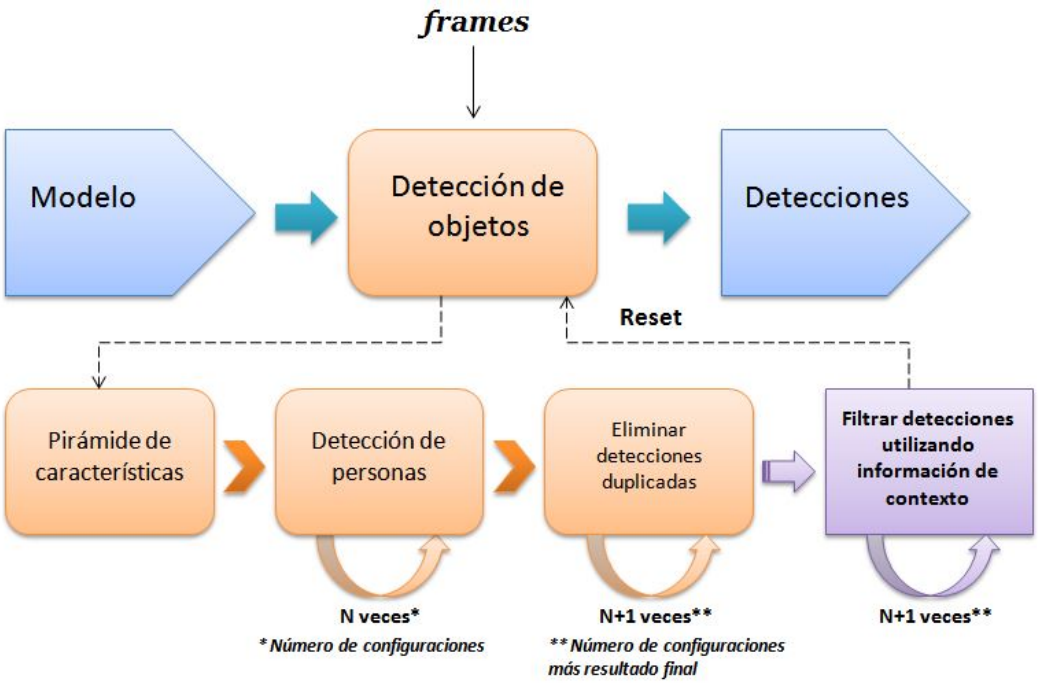


Figura 3.6: Esquema del detector implementado. La información de contexto se ha incorporado en un nuevo paso, representado en el esquema con los elementos de color morado.

varias modificaciones. Dentro del desarrollo de la implementación, hemos definido tres puntos clave:

- **Lectura de los mapas de confianza de oclusión:** al igual que leemos el fichero que contiene el modelo, hemos creado un método denominado *readconfidencemaps*. La función de este método es leer los mapas de confianza de oclusión obtenidos en la etapa anterior del sistema. Señalar que la lectura sólo se realiza una vez, no siendo determinante en el tiempo de ejecución final del sistema.

- **Método para filtrar las detecciones:** una vez obtenidos los *arrays* de localizaciones y puntuaciones tras la eliminación de las detecciones duplicadas, debemos filtrar las detecciones resultantes con los mapas de confianza. Para ello, el primer paso será obtener el nivel de escala del mapa de características correspondiente a cada detección. El detector original no proporcionaba este valor, por lo que lo añadimos como parte de la información asociada a cada detección.

  Teniendo el nivel de escala correspondiente a cada detección, podemos buscar el mapa de confianza relacionado con cada detección. El método implementado para eliminar las detecciones que no sean válidas acorde con el contexto se basa en una función lógica $XNOR$. El funcionamiento de la misma consiste en validar los *arrays* iguales y rechazar los diferentes. En nuestro caso, comparando los centros de cada *blob* para cada configuración del parte del cuerpo y su respectivo mapa de confianza. Cada configuración tiene un *array* adyacente de nueve posiciones, en el que cada posición corresponde a cada uno de los filtros. Si el filtro de la parte del cuerpo está activado, para esa configuración el valor es un 1; mientras que si no se utiliza, es un 0. Habiendo umbralizado los mapas de confianza de oclusión con valores de 1 ó 0, extraeremos el valor del mapa en la posición central del *blob* para los nueve filtros. De esta manera tendremos dos *arrays* de nueve posiciones y mediante la función $XNOR$ eliminaremos las detecciones que no sean posibles por la información de contexto.

- **Cálculo de los puntuaciones finales:** Cómo comentábamos en la sección 3.3.1, en el detector DTDP multiconfiguraciones [4] la puntuación varía según la disposición de cada configuración. No presenta la misma confianza detectar una persona completa, que únicamente una cabeza. Por ello, equiparamos las puntuaciones finales, ubicándolas en un mismo rango; permitiéndonos así evaluar la salida final del detector de manera adecuada.

Tras modificar los algoritmos existentes, tenemos implementado el sistema para

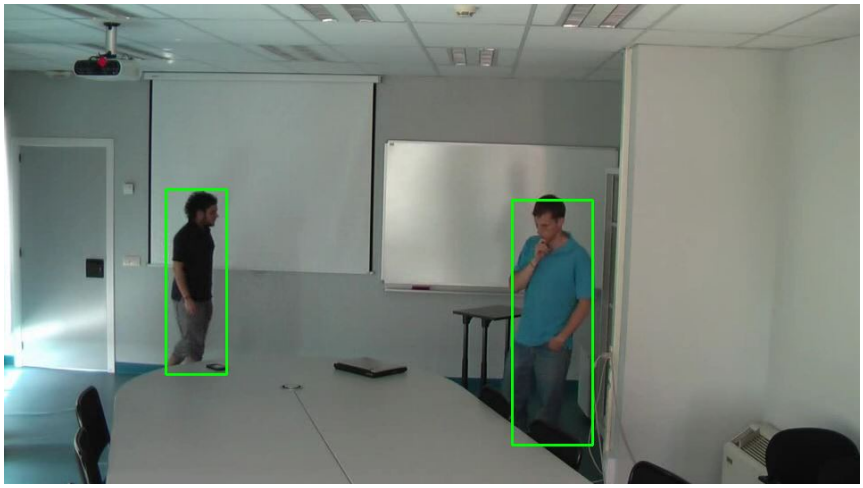


Figura 3.7: Ejemplo de representación de las detecciones utilizando información de contexto mediante la función *cvRectangle* para un *frame* de la *dataset* EDds (*Event Detection dataset*) [11].

detectar personas utilizando información de contexto. Con vistas a analizar el efecto producido por las mejoras introducidas, en el siguiente capítulo compararemos los resultados obtenidos, figura 3.7, con los logrados por el algoritmo disponible Latent SVM.

Capítulo 4

Evaluación del sistema

Una vez que tenemos el sistema desarrollado al completo, vamos a evaluar los resultados obtenidos. Para ello, compararemos el rendimiento de nuestro sistema con el del algoritmo DTDP [3].

En este capítulo, describiremos el proceso de evaluación propuesto en [12], así como se analizarán los resultados obtenidos por los dos algoritmos. Por este motivo, en la primera parte del capítulo explicaremos la elección de la *dataset*, el *ground truth*, el criterio de evaluación y las métricas de evaluación; dejando para el final el análisis de los resultados obtenidos.

4.1. *Dataset* y *Ground truth*

Dentro del proceso de evaluación, el primer paso es tener una base de datos de vídeos donde evaluar los resultados. En nuestro caso se ha elegido la *dataset* LIRIS [13] por dos razones. La primera es la naturaleza de los vídeos, ya que contienen los dos casos principales para los que hemos implementado el sistema:

- **Presencia de problemas de escala:** vídeos en los que las personas a detectar no se encuentran completas, quedando partes del cuerpo fuera de la resolución de la escena.
- **Presencia de oclusiones:** vídeos donde hay objetos que ocluyen parte de las personas, como mesas, o zonas donde sabemos que no podemos encontrar personas, como techos.

La segunda razón es por el *ground truth* que se proporciona con la *dataset*. El *ground truth* son los ficheros de anotación de la definición óptima de las personas mediante un rectángulo (*bounding box*) para cada uno de los *frames* de la secuencia. Los *bounding*

boxes se definen con los valores x, y, *width* y *height*. La posición (x, y) se corresponde con el vértice superior izquierdo del rectángulo, pudiendo calcular la posición de los otros vértices con los demás valores.

En este trabajo, para la evaluación se han seleccionado diez vídeos representativos de la *dataset* LIRIS [13]. Con las anotaciones correspondientes al *ground truth* y el criterio de evaluación explicado en la siguiente sección, podremos calcular las métricas de evaluación.

4.2. Criterio de evaluación

En esta sección vamos a describir el criterio de evaluación utilizado para medir el rendimiento de los algoritmos de detección de personas. Hemos escogido el criterio expuesto en [12], ya que para evaluar los resultados utilizaremos un sistema que utiliza este criterio del laboratorio de investigación VPULab, *Video Processing and Understanding Lab*.

Antes de entrar en detalles del criterio de evaluación, debemos definir las clasificaciones que obtendremos de la comparación entre los *bounding boxes* de los algoritmos de detección y los anotados en el *ground truth*. En función de las distintas combinaciones posibles entre las hipótesis arrojadas por los algoritmos y las anotaciones del *ground truth* se realizarán las siguientes clasificaciones:

- ***True Positive* (TP):** el sistema detecta personas y acierta.
- ***True Negative* (TN):** el sistema no detecta personas y acierta.
- ***False Positive* (FP):** el sistema detecta personas y se equivoca.
- ***False Negative* (FN):** el sistema no detecta personas y se equivoca.

El sistema que vamos a utilizar, no evalúa solamente si la hipótesis de una detección es correcta o no, sino también precisa la localización y extensión de las personas. Por esta razón, se va a utilizar el criterio de evaluación propuesto por [14]. Este criterio permite comparar hipótesis mediante tres estimaciones: distancia relativa (*relative distance*), *cover* y *overlap*.

La distancia relativa mide la distancia entre los centros de los *bounding boxes* en relación con el tamaño del *bounding box* anotado. *Cover* (área de los *bounding boxes* que coincide) y *overlap* (solape entre los *bounding boxes*) estiman el porcentaje del *bounding box* anotado que es cubierto por la hipótesis de detección y viceversa. Una detección es considerada como válida si la distancia relativa es menor o igual a 0.5 (correspondiendo a una desviación mayor al 25 % del verdadero tamaño del objeto),

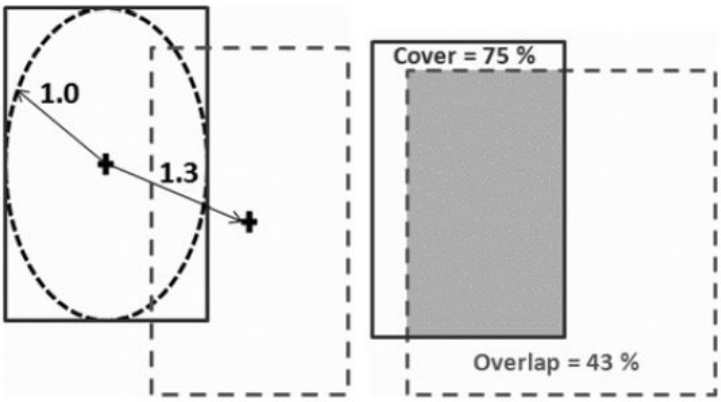


Figura 4.1: Criterio de evaluación para la comparación de *bounding boxes*. La imagen izquierda corresponde a la distancia relativa y la derecha al *cover* y *overlap*. Imagen extraída de [12].

y *cover* y *overlap* están ambos por encima del 50 %. Únicamente es aceptada como correcta una hipótesis por persona, por lo que cualquier hipótesis adicional de la misma persona es considerada como un falso positivo (FP). En la figura 4.1 podemos observar un ejemplo visual de las tres estimaciones propuestas por [14].

4.3. Métricas de evaluación

Para evaluar el rendimiento de nuestro sistema de detección de personas es necesario cuantificar el rendimiento, comparándolo con el del detector DTDP [3] existente en el estado del arte. Una vez clasificadas las hipótesis de detección según el criterio de evaluación, podemos medir el rendimiento. Este se puede evaluar en secuencias completas mediante las curvas *precision-recall*. Las salidas de los algoritmos de detección son valores de confianza (*scores*), donde los valores más altos indican una mayor probabilidad de que la hipótesis de detección sea correcta. El método de evaluación que vamos a utilizar compara los parámetros de *precision* y *recall* desde los valores de confianza más bajos a los más altos. Los valores de *precision* y *recall* se pueden calcular de la siguiente manera:

$$precision = \frac{\#TP}{\#TP+\#FP} \tag{4.1}$$

$$recall = \frac{\#TP}{\#TP+\#FN} \tag{4.2}$$

Con las ecuaciones 4.1 y 4.2 obtenemos para cada valor de confianza (*score*) un

punto de la curva *precision-recall*. El análisis de esta curva se resume en el aumento del valor de *precision* cuando hay pocos falsos positivos, y incremento del valor de *recall* cuando hay pocos falsos negativos. Calculando el área bajo la curva (AUC-PR) de cada una de ellas obtendremos un valor discriminante, donde a mayor área bajo la curva, mejor funcionamiento del sistema.

4.4. Análisis y resultados

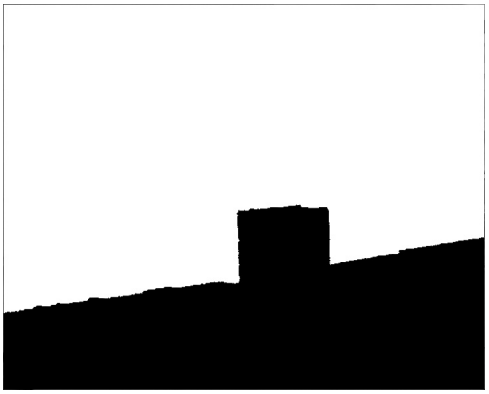
Una vez explicado el método de evaluación, vamos a analizar los resultados obtenidos. Hemos calculado la curva *precision-recall* de diez vídeos de la dataset LIRIS [13] para nuestro sistema y el detector DTDP [3], proporcionándonos una idea de la mejora del rendimiento. Se han evaluado nuestro sistema implementado y el detector DTDP original [3], en lugar del algoritmo DTDP multiconfiguraciones. Esto es debido a que el detector DTDP multiconfiguraciones tiene como objetivo la detección de personas en grupos, siendo lo habitual la utilización del DTDP base para secuencias con las características a mejorar en este trabajo.

Con la curva *precisión-recall*, como con el área bajo la curva (AUC-PR) podremos analizar si la utilización de información de contexto mejora el rendimiento. Primero, analizaremos la curva *precision-recall* para una secuencia representativa; después, expondremos los resultados del área bajo la curva para todas las secuencias y por último, estudiaremos el resultado de un caso concreto.

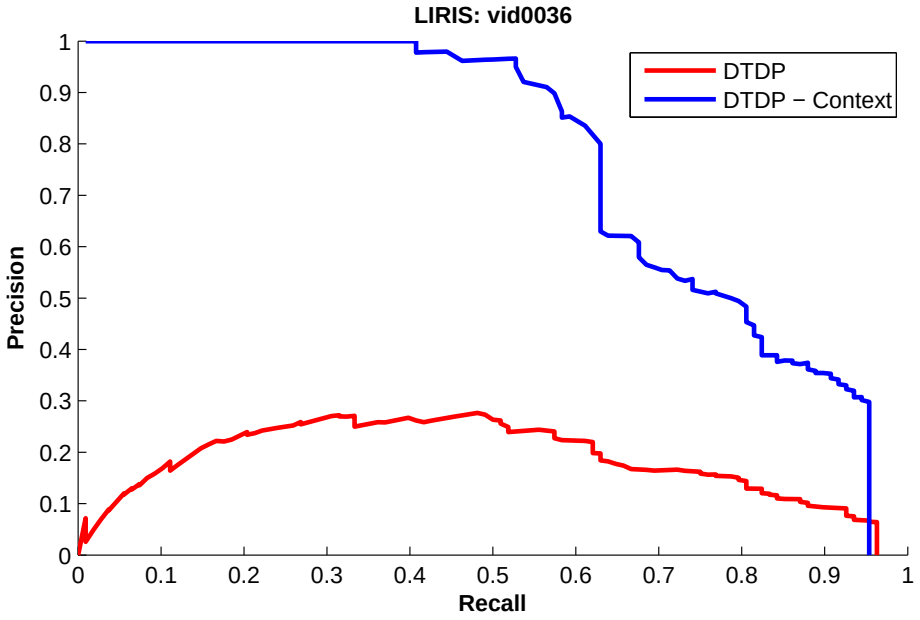
Dentro de los diez vídeos escogidos de la *dataset* LIRIS, vamos a analizar los resultados de una secuencia representativa. En la figura 4.2 podemos observar los resultados de la evaluación para el vídeo 'vid0036'. La escena presenta una mesa, considerada como oclusiva, y el objetivo es detectar a una persona guardando un libro. En la figura 4.2 (c) vemos la curva *precision-recall*. Esta presenta una gran mejora, siendo significativo el aumento de la *precision*. En cuanto al *recall*, el incremento también es relevante, aunque se pierde un poco para valores cercanos al caso óptimo. En algunos de los vídeos también ocurre esta situación con el *recall*, no siendo especialmente importante si tenemos en cuenta el compromiso entre la mejora total del sistema y el *recall* para valores altos. Para finalizar el análisis de esta secuencia, señalar que podemos afirmar que la utilización de información de contexto supone una mejora en el rendimiento del detector DTDP [3].

Para poder medir el rendimiento, vamos a calcular el área bajo la curva *precision-recall* (AUC-PR). Ya hemos observado una de ellas, mientras que las demás curvas *precision-recall* se pueden encontrar en el apéndice A de este trabajo. En la tabla 4.1 podemos observar el valor del área bajo la curva para el detector DTDP y nuestro

(a)



(b)



(c)

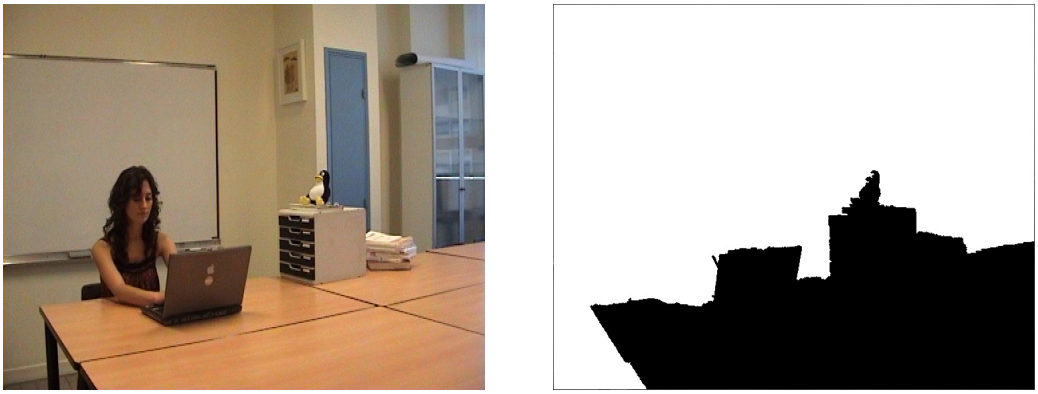
Figura 4.2: Resultados de la evaluación de la secuencia 'vid0036' de la *dataset* LIRIS [13].
(a) *Frame* representativo: una persona a detectar
(b) Imagen con las zonas oclusivas anotadas: mesa
(c) Curva *precision-recall*

Vídeo de la *dataset* LIRIS	**Área bajo la curva *precision-recall* (AUC-PR)**		**%Δ**
	DTDP	**DTDP-Context**	
'vid0001'	75.90	86.95	14.56
'vid0003'	63.63	94.86	49.08
'vid0036'	18.32	76.45	317.30
'vid0040'	92.75	89.91	-3.06
'vid0044'	11.64	69.53	497.34
'vid0049'	28.82	68.85	138.90
'vid0061'	39.49	66.95	69.54
'vid0068'	80.55	87.52	8.65
'vid0081'	12.57	85.85	582.98
'vid0090'	64.64	92.91	43.73

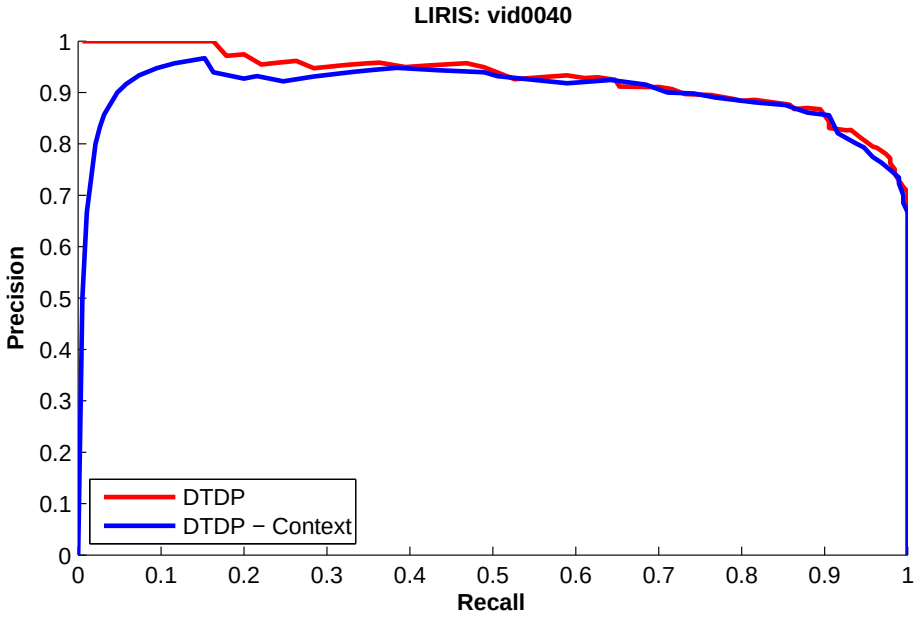
Tabla 4.1: Resultados obtenidos de la evaluación de diez vídeos de la *dataset* LIRIS [13]. En la tabla se muestra el valor del área bajo la curva (AUC-PR) para el detector DTDP [3] y nuestro sistema, denominado DTDP-Context. Para comparar los rendimientos, se ha calculado el incremento porcentual **%Δ.**

sistema, denominado DTDP-Context, así como el incremento porcentual. Es de destacar que el incremento porcentual es positivo en nueve de los diez vídeos. Se puede observar que la utilización de información de contexto aumenta el área bajo la curva de manera significativa para los vídeos donde el detector DTDP ofrece valores bajos. Por otro lado, cuando el detector DTDP ofrece buenos resultados, aún siendo más complicado mejorar el rendimiento, también conseguimos mejoras notables.

Tras comprobar que la utilización de información de contexto mejora el rendimiento del detector DTDP, vamos a analizar un caso concreto. En la figura 4.3 podemos observar la única secuencia en la que el área bajo la curva disminuye. Este hecho se puede explicar por dos razones: el gran rendimiento del detector DTDP y la localización del centro del *bounding box* en la zona oclusiva. La primera causa se puede interpretar como la dificultad que supone mejorar un detector que obtiene resultados casi óptimos, aunque la disminución porcentual sólo es 3.06. En cuanto a la segunda, se debe a la características propias de la secuencia, donde el centro de la persona se localiza dentro de la mesa, anotada como oclusiva. En este caso, los brazos, las sombras, la mesa y el portátil presentan bordes, que el detector original DTDP interpreta como algo que podría parecerse a unas piernas. Este hecho hace que se detecte a la persona con más confianza que en nuestro sistema, ya que el valor de confianza es menor debido a que únicamente detectamos la configuración correspondiente a medio cuerpo. Por este motivo, la curva *precision-recall* pierde *precision*, debido a que en nuestro sistema se detecta a la persona pero con un valor de confianza menor.



(a) (b)



(c)

Figura 4.3: Resultados de la evaluación de la secuencia 'vid0040' de la *dataset* LIRIS [13].
(a) *Frame* representativo: una persona a detectar
(b) Imagen con las zonas oclusivas anotadas: mesa
(c) Curva *precision-recall*

Capítulo 5

Conclusiones y trabajo futuro

5.1. Conclusiones

Este trabajo tenía como propósito la incorporación de información de contexto al detector DTDP [3] existente en el estado del arte. Para cumplir con el objetivo se ha implementado un sistema que permite anotar las zonas oclusivas de la escena, obtener los mapas de confianza de oclusión y detectar personas utilizando la información de contexto.

Para llegar a este fin, se llevó a cabo un estudio de los diferentes algoritmos del estado del arte y su posterior desarrollo. Para la etapa de segmentación, se estudió el algoritmo *Watershed* [1] y se creó un sistema de anotación de zonas oclusivas sencillo e intuitivo. El siguiente paso fue analizar el método de obtención de mapas de confianza [2], adaptando e implementando el algoritmo a C++. Por último, se estudiaron en profundidad los algoritmos de detección de personas existentes, incorporando al DTDP multiconfiguraciones [4] la información de contexto. De esta forma, se evita tener que reentrenar el modelo de persona para secuencias con problemas de escala o de oclusiones.

Por último se ha estudiado el criterio de evaluación para los detectores de personas y se ha evaluado el rendimiento del sistema implementado. Los resultados obtenidos han dado veracidad a la hipótesis inicial, obteniendo notables mejoras en el rendimiento comparando con los resultados del detector DTDP.

5.2. Trabajo futuro

Partiendo del trabajo realizado, es evidente la necesidad de seguir trabajando en algoritmos de detección de personas, mejorando su rendimiento en condiciones complejas. En concreto para el sistema implementado se proponen varias líneas de investigación como parte del trabajo futuro.

El detector DTDP [3] en el que se basa este trabajo es un algoritmo lento, razón por la cual no ha sido posible ejecutar nuestro sistema sobre vídeos en tiempo real. Por este motivo, una de las líneas de investigación propuesta como trabajo futuro es el desarrollo de un algoritmo más rápido, limitando la zona de búsqueda espacial ya que se trata de un algoritmo de búsqueda exhaustiva. Otra opción puede ser limitar el rango de escalas a analizar de forma automática, permitiendo de esta forma calcular menos mapas de confianza de oclusión.

Como mejora del trabajo realizado también se propone la adaptación del sistema a vídeos de cámaras no estáticas. Muchas de las cámaras de vídeo-seguridad son móviles, pudiendo enfocar a un entorno controlado. Para este caso, se podría adaptar el sistema, creando una panorámica con las zonas oclusivas anotadas para las diferentes escenas posibles.

Aunque en este trabajo no se han analizado este tipo de algoritmos se propone el estudio de algoritmos de seguimiento o *tracking*, que son más robustos a oclusiones y permiten seguir a múltiples objetivos. Con la incorporación de información de movimiento se cree probable que se mejore el rendimiento y se añada robustez a la detección.

Bibliografía

[1] S. Beucher and C. Lantuéjoul, "Use of watersheds in contour detection," in *Proceedings of International Workshop of Image Processing, Real-time edge motion detection*, 1979.

[2] A. García-Martín and J. C. SanMiguel, "Context-aware part-based people detection for video monitoring," *Electronics Letters*, vol. 51, no. 23, pp. 1865–1867, 2015.

[3] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[4] A. García-Martín, R. H. Evangelio, and T. Sikora, "Multi-configurations for part-based person detectors," in *Proceedings of ECCV*, 2014.

[5] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2005. CVPR 2005.*, vol. 1, pp. 886–893, IEEE, 2005.

[6] OpenCV, "http://www.opencv.org/."

[7] Matlab, "http://es.mathworks.com/."

[8] N. L. Serna-Palomino and L. Pro-Concepción, "Watershed: an algorithm efficient and flexible segmentation of footage gels 2-of," *Revista de investigación de Sistemas e Informática*, vol. 7, no. 2, 2014.

[9] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.

[10] J. C. SanMiguel and J. M. Martínez, "An ontology for event detection and its application in surveillance video," pp. 220–225, 2009.

[11] J. C. SanMiguel, M. Escudero-Vinolo, J. M. Martínez, and et al., "Real-time single-view video event recognition in controlled environments," pp. 91–96, 2011.

[12] A. García-Martín and J. M. Martínez, "People detection in surveillance: classification and evaluation," *IET Computer Vision*, vol. 9, no. 5, pp. 779–788, 2015.

[13] C. Wolf, J. Mille, O. Celiktutan, M. Jiu, E. Dogan, G. Eren, M. Baccouche, E. Dellandrea, C.-E. Bichot, C. Garcia, and B. Sankur, "Evaluation of video activity localizations integrating quality and quantity measurements," vol. 127, no. 1, pp. 14–30, 2014.

[14] B. Leibe, E. Seemann, and B. Schiele, "Pedestrian detection in crowded scenes," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, CVPR '05, (Washington, DC, USA), pp. 878–885, IEEE Computer Society, 2005.

Glosario

AUC *Area Under Curve*

BLOB *Binary Large Object*

DTDP *Discriminatively Trained Deformable Parts*

EDds *Event Detection Datasets*

FN *False Negative*

FP *False Positive*

HOG *Histogram of Oriented Gradients*

LIRIS *Laboratoire d'InfoRmatique en Image et Systèmes d'information*

OpenCV *Open Source Computer Vision*

ROI *Region Of Interest*

SVM *Support Vector Machine*

TN *True Negative*

TP *True Positive*

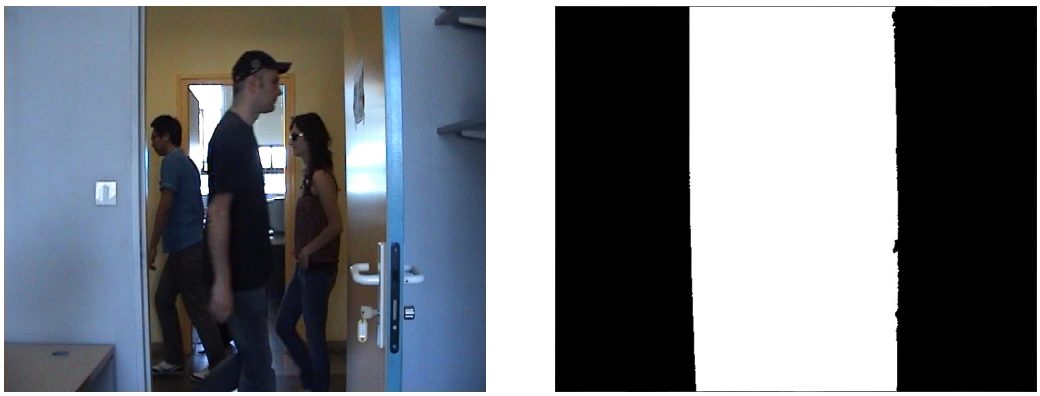
VPULab *Video Processing and Understanding Lab*

Apéndice A

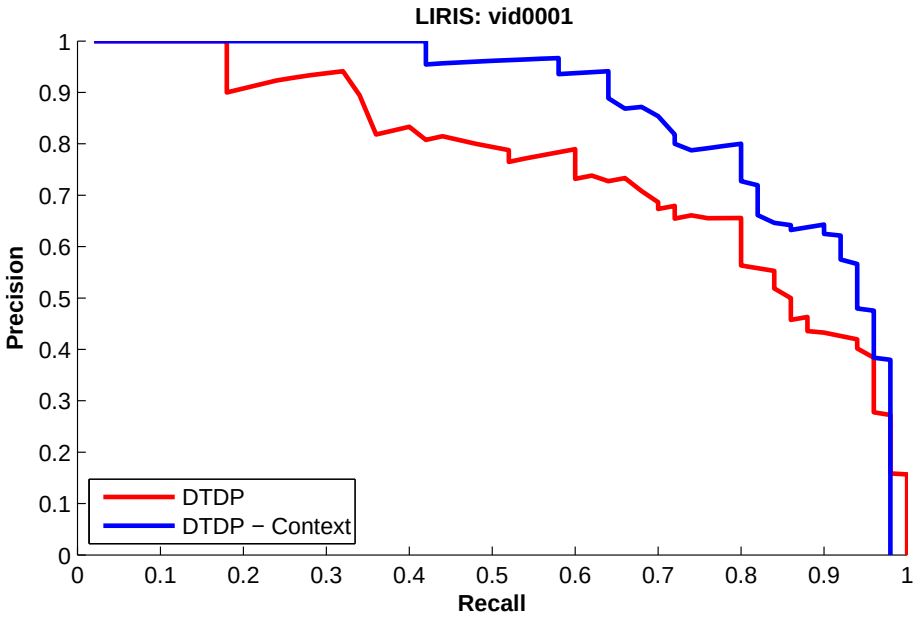
Curvas *precision-recall* para las secuencias de la *dataset* LIRIS

Este apéndice contiene las curvas *precision-recall* para los vídeos seleccionados de la *dataset* LIRIS [13]. Para poder analizar los resultados, se muestra un *frame* representativo de cada secuencia y la imagen con las zonas oclusivas anotadas. Las objetos estáticos oclusivos están representados con el color negro, mientras que el fondo de la escena está definido con el blanco. Señalar que en algunos de los vídeos esta imagen es blanca, pues la secuencia no presenta oclusiones, sino problemas de escala. Estos problemas se pueden apreciar en los *frames* correspondientes, donde hay partes de las personas a detectar que no aparecen en la resolución de la secuencia.

A.1. Vídeo 'vid0001'



(a) (b)

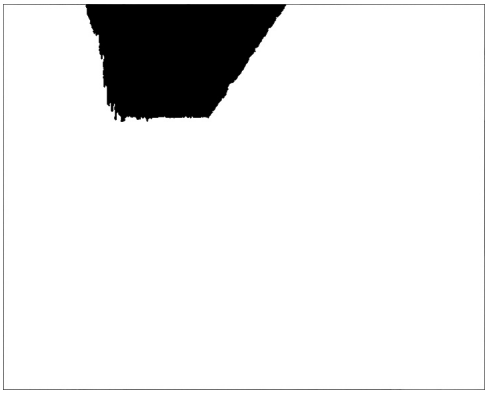


(c)

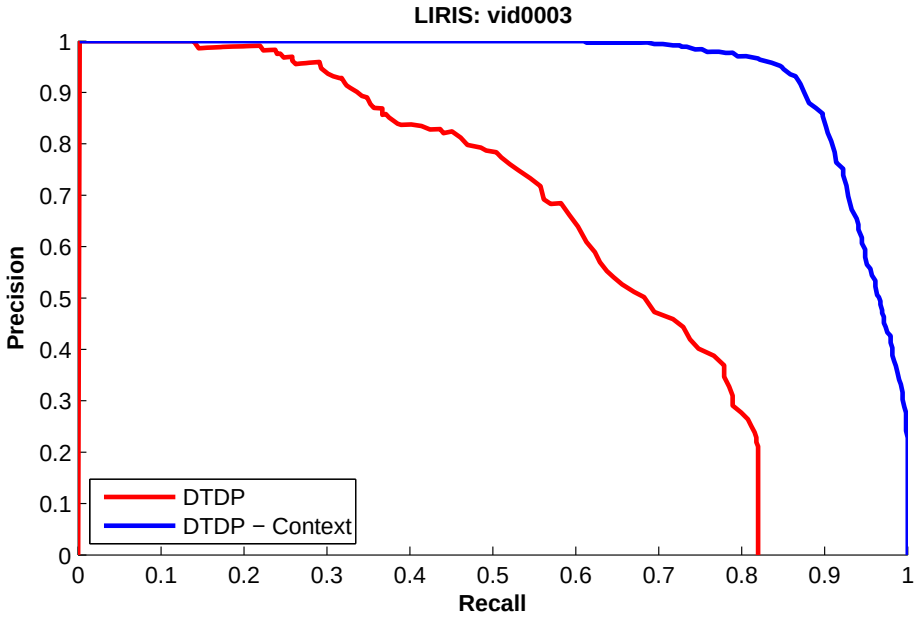
Figura A.1: Resultados de la evaluación de la secuencia 'vid0001' de la *dataset* LIRIS [13].
(a) *Frame* representativo: tres personas a detectar
(b) Imagen con las zonas oclusivas anotadas: laterales de la escena
(c) Curva *precision-recall*

A.2. Vídeo 'vid0003'

(a)



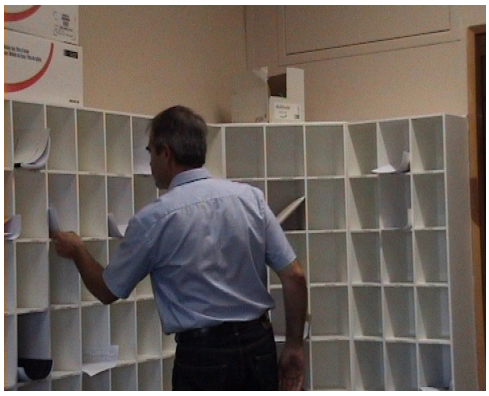
(b)



(c)

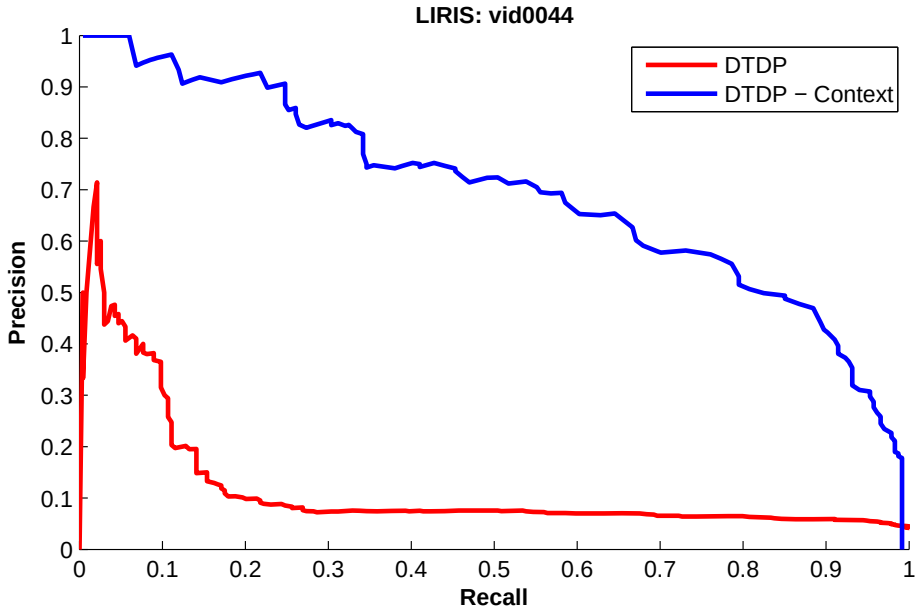
Figura A.2: Resultados de la evaluación de la secuencia 'vid0003' de la *dataset* LIRIS [13].
(a) *Frame* representativo: dos personas a detectar
(b) Imagen con las zonas oclusivas anotadas: techo
(c) Curva *precision-recall*

A.3. Vídeo 'vid0044'



(a)

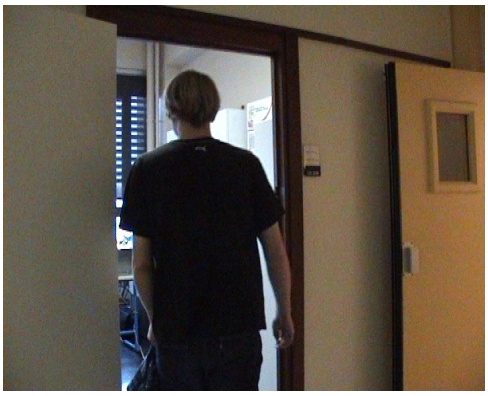
(b)



(c)

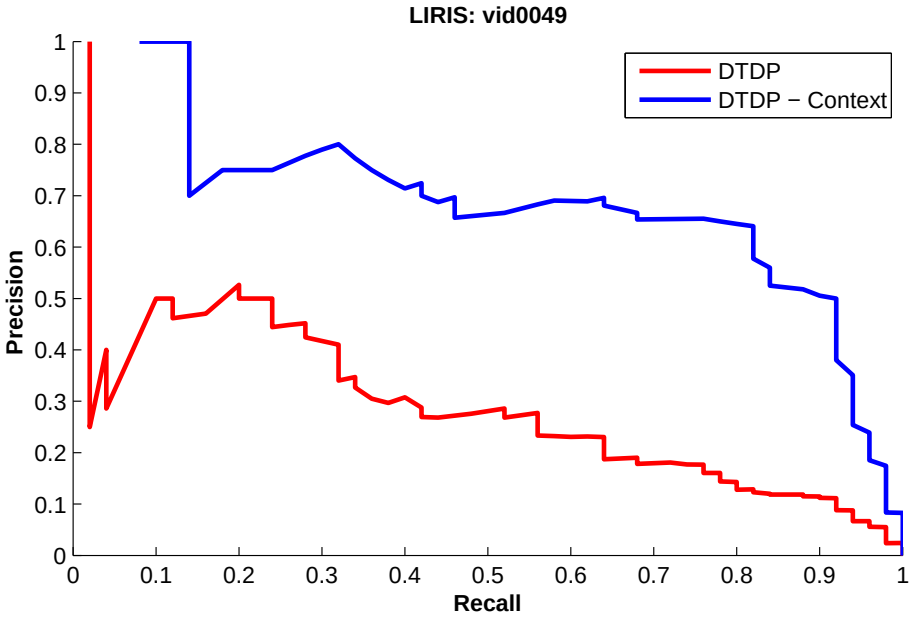
Figura A.3: Resultados de la evaluación de la secuencia 'vid0044' de la *dataset* LIRIS [13].
(a) *Frame* representativo: una persona a detectar
(b) Imagen con las zonas oclusivas anotadas: sin anotación, la secuencia presenta problemas de escala
(c) Curva *precision-recall*

A.4. Vídeo 'vid0049'



(a)

(b)

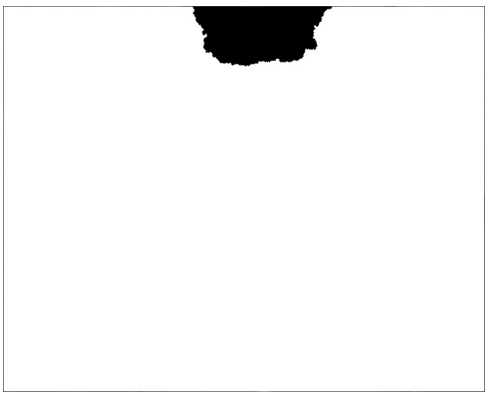


(c)

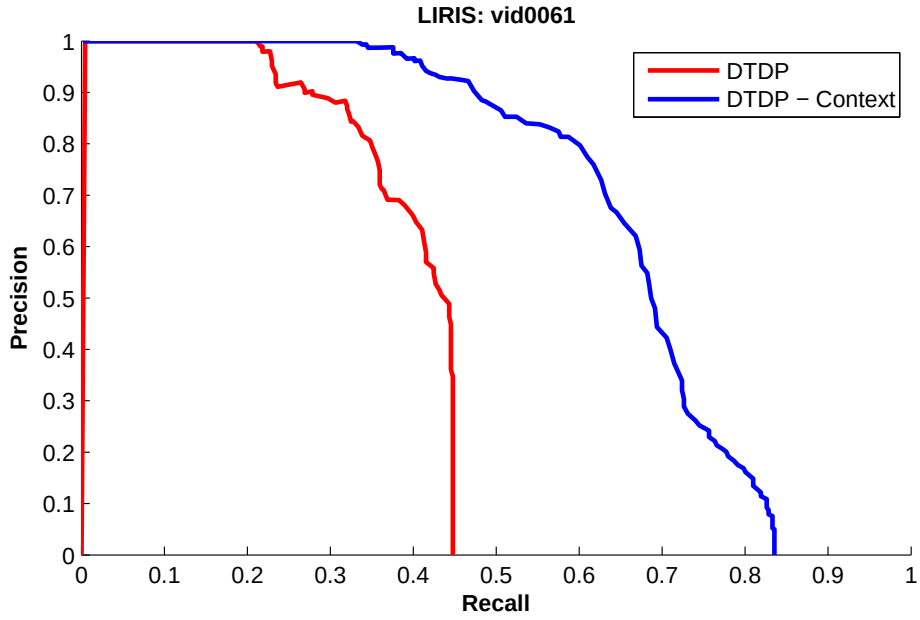
Figura A.4: Resultados de la evaluación de la secuencia 'vid0049' de la *dataset* LIRIS [13].
(a) *Frame* representativo: una persona a detectar
(b) Imagen con las zonas oclusivas anotadas: sin anotación, la secuencia presenta problemas de escala
(c) Curva *precision-recall*

A.5. Vídeo 'vid0061'

(a)



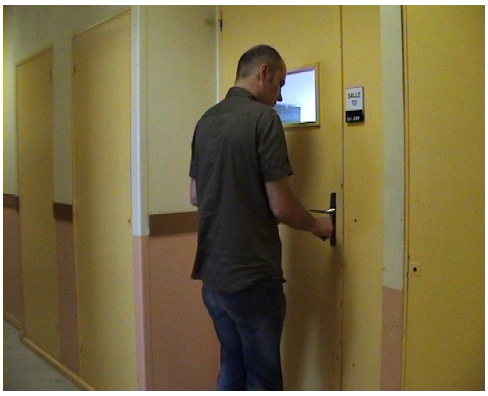
(b)



(c)

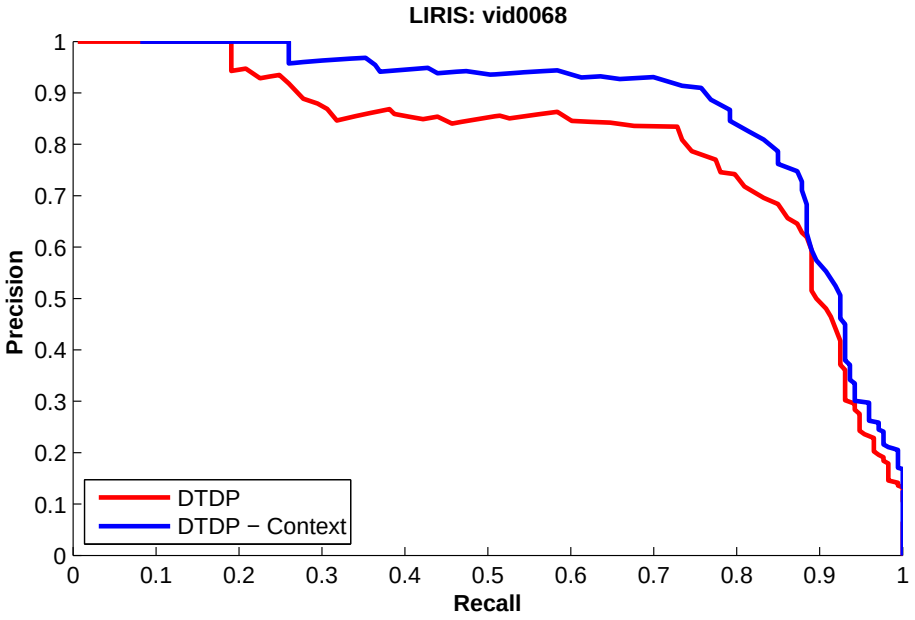
Figura A.5: Resultados de la evaluación de la secuencia 'vid0061' de la *dataset* LIRIS [13].
(a) *Frame* representativo: dos personas a detectar
(b) Imagen con las zonas oclusivas anotadas: techo
(c) Curva *precision-recall*

A.6. Vídeo 'vid0068'



(a)

(b)

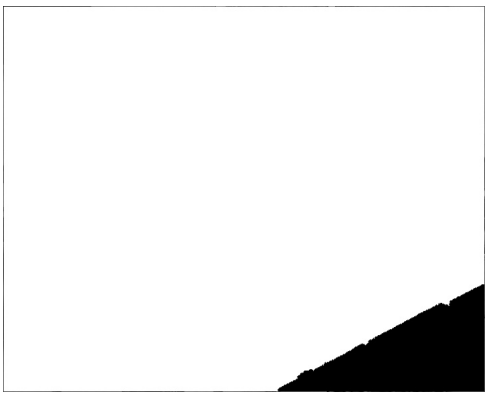


(c)

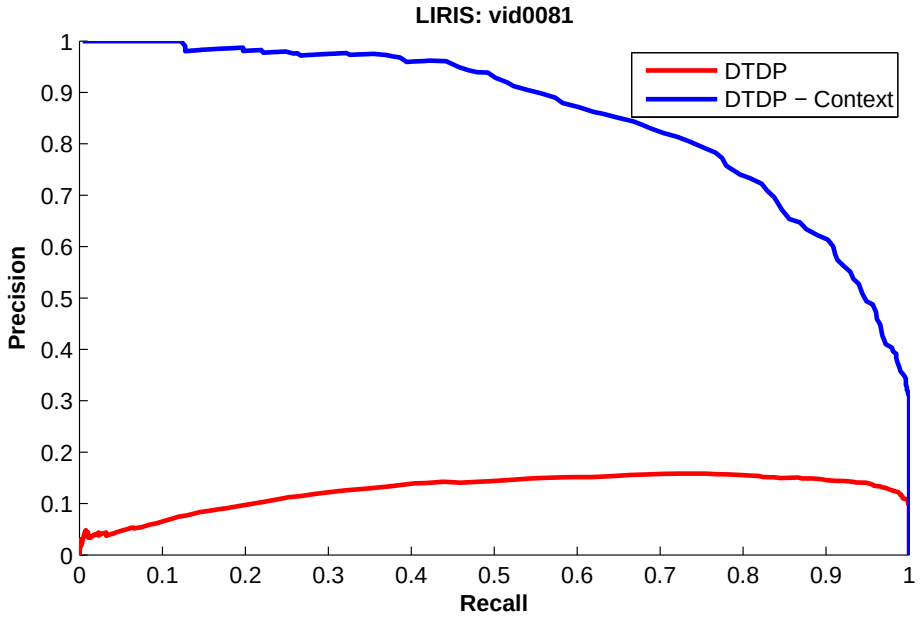
Figura A.6: Resultados de la evaluación de la secuencia 'vid0068' de la *dataset* LIRIS [13].
(a) *Frame* representativo: una persona a detectar
(b) Imagen con las zonas oclusivas anotadas: sin anotación, la secuencia presenta problemas de escala
(c) Curva *precision-recall*

A.7. Vídeo 'vid0081'

(a)



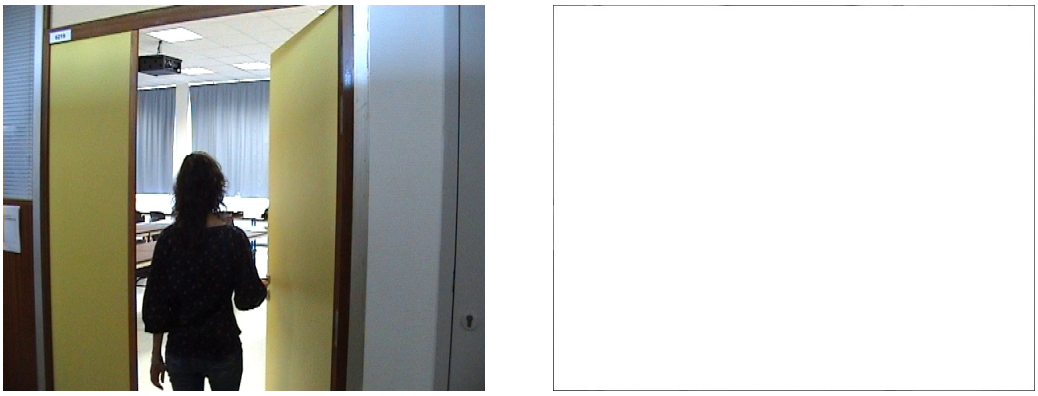
(b)



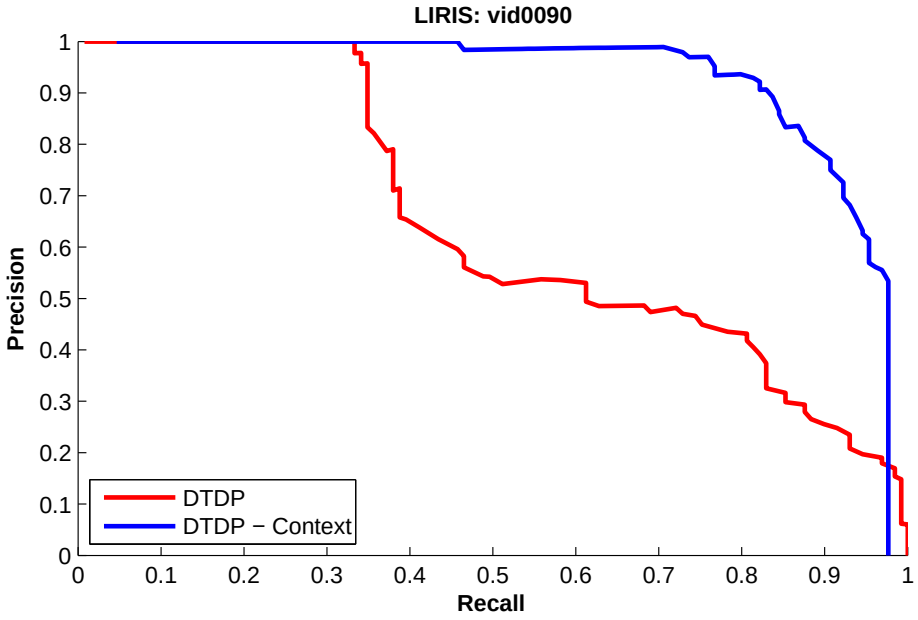
(c)

Figura A.7: Resultados de la evaluación de la secuencia 'vid0081' de la *dataset* LIRIS [13].
(a) *Frame* representativo: dos personas a detectar
(b) Imagen con las zonas oclusivas anotadas: mesa en la esquina inferior derecha
(c) Curva *precision-recall*

A.8. Vídeo 'vid0090'



(a) (b)



(c)

Figura A.8: Resultados de la evaluación de la secuencia 'vid0090' de la *dataset* LIRIS [13].
(a) *Frame* representativo: una persona a detectar
(b) Imagen con las zonas oclusivas anotadas: sin anotación, la secuencia presenta problemas de escala
(c) Curva *precision-recall*